In The Name of GOD

Sharif University of Technology
Electrical Engineering Department

Deep Learning
Final Project

GAN_BERT

Pouria Dadkhah
Armin Ghoujezadeh
Atieh Mirzaie

Supervisor
Dr. Fatemizadeh

February 2024

Context

Chapter 1

Introduction

Today, many deep learning methods have been used in natural language processing. These from methods use transformer-like structures. BERT [1] is used to calculate the appropriate representation for the data. These structures are basically trained on a large set of data and then fine-tuned for different applications to reach higher accuracy. This high accuracy is achieved when these models are trained on thousands of labeled data.

Achieving this amount of label income is a complicated and time-consuming process; For this reason, semi-observation methods have expanded a lot. In these methods, it is assumed that we have limited labeled data available, but there is also access to unlabeled data. One of these structures SS-GAN There are two parts, the generator and the purifier. The generative network is used to produce samples similar to the original data and the discriminating network is used to distinguish the generative samples from the actual training data. Of course, the discriminator, in addition to identifying fake or genuine samples, also classifies them in several classes.

In continuation SS-GAN, the structure GAN-BERT [2], which is a generalization of the modelBERTIn the generator mode, it is semi-observed, and we also explain the data used in this project.

1-1SS-GAN

Discriminator in SS-GAN on the k+1 They see the training class: examples in the classrooms from 1,2,...,k And fake samples k+1 Classes are classified. In other words, supposed *p* and *p_G* According to the probability distribution of real and fake samples. Also assume

$$p_m(\hat{y} = y|x, y = k + 1)$$

An indicator of the possibility of the sample being fake *x* and

$$p_m(\hat{y} = y|x, y \in (1, ..., k))$$

indicator The probability of the sample being true (which as a result belongs to one of *k* Be first class (for the model *m* be In this case, the cleaning cost function is defined as follows:

$$L_D = L_{D_{sup}} + L_{D_{unsup}}$$

in which we have:

$$L_{D_{sup}} = -\mathbb{E}_{x,y\sim p_d} \log[p_m(\hat{y} = y|x, y \in (1, ..., k))]$$

$$L_{D_{unsup}} = -\mathbb{E}_{x\sim p_d} \log[1 - p_m(\hat{y} = y|x, y = k + 1)] - \mathbb{E}_{x\sim G} \log[p_m(\hat{y} = y|x, y = k + 1)]$$

*supdL*The error of assigning the sample to the wrong class (which is one of*k*It is first class*unsupdL*It also considers the error of identifying a valid sample without a label as a fake sample.

> At the same time, it is expected that it will be born at nightGBe able to make samples similar to real samples distribution

*dp*produce In other words, the average of the samples produced by the generator should be closetothetoamount the real data. assume)*x*(*f*The output indicator of the activity function in the middle*d*be; In this case,Fromtheitlayers depends on the costfeature matchingforGIt is defined as follows:

$$L_{G_{feature\ matching}} = ||\mathbb{E}_{x\sim p_d} f(x) - \mathbb{E}_{x\sim G} f(x)||_2^2$$

It is also necessary to consider the delay cost function for the generator to calculate the error caused by the detection

of fake data by the discriminator:

$$L_{G_{unsup}} = -\mathbb{E}_{x \sim G} \log[1 - p_m(\hat{y} = y | x, y = k + 1)]$$

In this way, the cost function of the generator is equal to

$$L_G = L_{G_{feature\ matching}} + L_{G_{unsup}}.$$

1-2 GAN-BERT

in the structure GAN-BERT which in 1-1Shown, cleaned The task of classifying each sample into one *k* has a class. It should also identify whether the data is real or generated by the generator generally k+1 class G It also has the task of producing a display similar to the display of actual data. The structure of each of these two nights is shown in Figure 1-2.

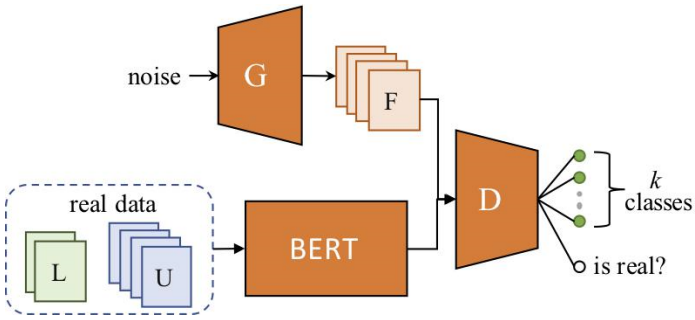


Figure 1-1: Architecture GAN-BERT

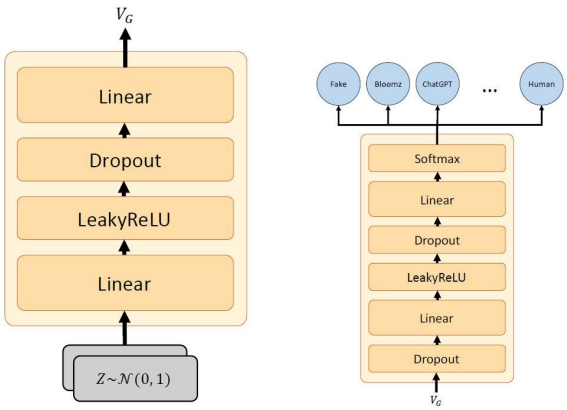


Figure 1-2: network architecture D It is the right side *G_1* left side

You can also use a late architecture for the generator, which is in the shell.1-3It has been shown. In this architecture, he was trained over night. BERT It is used independently of BERT Used in Shell1-1 it is In this way, first bag of words It is made from the words present in the training samples. Then, based on the distribution of these words, random samples *X* in the form id Selection and to
BERT are given

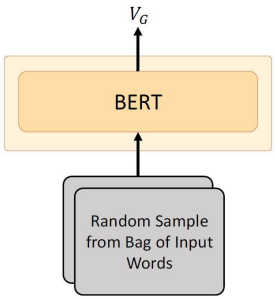


Figure 1-3: Network architecture *G_2*

1-3 Dataset

This data project is a generalization of data M4 Its specifications are in the table1-1 It has been shown.

This data is formattedjsonand each sample is composed of the following items:

- Id: Sample ID
- Label: Sample class (classes):(human, ChatGPT, Cohere, Davinci, Bloomz, Dolly
- Text: The text is going to be classified
- Model: The model by which the text was produced
- Source: The source of the obtained sample is includedWikipedia, Wikihow, Peerread, Reddit, Arxiv

Table 1-1: Data characteristics

Source/ Domain	Language	Total Human	Parallel Data						
			Human	Davinci003	ChatGPT	Cohere	Dolly-v2	BLOOMz	Total
Wikipedia	English	6,458,670	3,000	3,000	2,995	2,336	2,702	3,000	17,033
Reddit ELI5	English	558,669	3,000	3,000	3,000	3,000	3,000	3,000	18,000
WikiHow	English	31,102	3,000	3,000	3,000	3,000	3,000	3,000	18,000
PeerRead	English	5,798	5,798	2,344	2,344	2,344	2,344	2,344	17,518
arXiv abstract	English	2,219,423	3,000	3,000	3,000	3,000	3,000	3,000	18,000
Baike/Web QA	Chinese	113,313	3,000	3,000	3,000	–	–	–	9,000
RuATD	Russian	75,291	3,000	3,000	3,000	–	–	–	9,000
Urdu-news	Urdu	107,881	3,000	–	3,000	–	–	–	9,000
id_newspapers_2018	Indonesian	499,164	3,000	–	3,000	–	–	–	6,000
Arabic-Wikipedia	Arabic	1,209,042	3,000	–	3,000	–	–	–	6,000
True & Fake News	Bulgarian	94,000	3,000	3,000	3,000	–	–	–	9,000
Total			35,798	23,344	32,339	13,680	14,046	14,344	133,551

In this project k = 6 and the purpose of classification is based on the input text on each of the classes. For this purpose, we consider different percentages of the dataset as labeled data and gradually increase the number of these samples and compare and report the results.

Chapter 2

The first request

In this part of the model BERT with the aim of classifying texts into different classes fine-tune We will have a detailed explanation on how to implement this section and its results.

2-1 Explanation of the method

The implementation steps are as follows:

1. Calling libraries: In this section, there are various libraries, including items related to data. (using libraries Hugging Face) data changes (Pandasand, NumPyand) other libraries .(scikit-learn and Matplotlib) We will call.
2. Adjust seed And the device
3. Loading data and dividing it into sections test, train and validation
4. Preparation: from BERT tokenizer To tokenize the training data and use the credential meter.
5. value Basic model: stratified model based on BERT(Bert For Sequence Classification) value It becomes primary and the number of its labels is determined.

6. Determination of training hyperparameters: items including training rate, batch size, evaluation strategy and

   Maximum number of learning steps (max_step) will be appreciated. Trainer You will be comfortable with training data and reliability. Variable max_step It is the maximum number of steps (children) of education that fine-tune It will be implemented. This variable is used to limit the learning process, especially when the dataset is very large. In fact, this variable, in addition to regulating the training time, prevents overfiting It will also happen.

7. Model training with a certain number of epochs and storing the best model based on the results of the validation meter.

8. Showing the results: the same criteria recall, precision, accuracyandf1-scoreFor each class

   Calculated and displayed on the graph. The meanings of each of these criteria are as follows:

   - Accuracy: The ratio of the number of correctly classified samples to the total number of samples
   - Precision: The ratio of the number of samples that were correctly predicted in each class to the total number of samples that were predicted as members of that class.
   - Recall: The ratio of the number of samples that are correctly predicted in each class to the total number of samples that are actually in that class.
   - f1-score: average precision and recall

2-2 Results

We will check the results by considering 80 percent of the samples as the training set and 20 percent of them for the test (by increasing the training percentage, we will have a better result!). We also decide max_step= 50 The results are in this order:

```
There are 1 GPU(s) available.
We will use the GPU: NVIDIA GeForce RTX 3070 Laptop GPU
```

Figure 2-1: gpu

```
[11]: train_df
```

[11]:

	text	model	source	label	id
63795	We report on observations made with the Spitze...	bloomz	arxiv	4	63795
52761	The concept of dynamical 3-space is introduced...	bloomz	arxiv	4	52761
17690	The European Association for Digital Humaniti...	davinci	wikipedia	3	17690
38189	We study theoretically the emission and absorp...	bloomz	arxiv	4	38189
18742	Stranger in the City, also known as S.I.T.C, ...	davinci	wikipedia	3	18742
...	...	...	...	...	...
31671	The chiral condensate is an important order pa...	dolly	arxiv	5	31671
29500	Yes, mercenaries in the Middle Ages carried ba...	davinci	reddit	3	29500
25898	Well, that's an interesting question. Prester ...	chatGPT	reddit	1	25898
66736	Spend your time wisely. Stop comparing yoursel...	dolly	wikihow	5	66736
13891	Laboa (Basque pronunciation:[laβoβa]) is the n...	bloomz	wikipedia	4	13891

56821 rows × 5 columns

Figure 2-2: Teaching data

```
[12]: test_df
```

[12]:

	text	model	source	label	id
0	Overall, I found the paper "Machine Comprehens...	chatGPT	peerread	1	1844
1	This paper "Machine Comprehension Using Match-...	chatGPT	peerread	1	1845
2	The paper presents an end-to-end neural archit...	chatGPT	peerread	1	1846
3	This paper proposes an end-to-end neural archi...	chatGPT	peerread	1	1847
4	Title: Incorporating long-range consistency in...	chatGPT	peerread	1	1848
...	...	...	...	...	...
2995	The paper Energy-Based Spherical Sparse Coding...	dolly	peerread	5	14560
2996	Dear Author, I have reviewed your submitted pa...	dolly	peerread	5	14561
2997	Denoising Auto-Encoders (DAE) have been used i...	dolly	peerread	5	14562
2998	The paper Revisiting Denoising Auto-Encoders, ...	dolly	peerread	5	14563
2999	This paper Revisiting Denoising Auto-Encoders ...	dolly	peerread	5	14564

3000 rows × 5 columns

Figure 2-3: Test data

```
[13]: val_df
```

[13]:

	text	model	source	label	id
30773	It is really cool that there is no sound in a ...	davinci	reddit	3	30773
55574	\n\nChanging the water pump on a 2.0L, 4 Cylin...	davinci	wikihow	3	55574
39797	In our work titled "Random Access Broadcast: S...	chatGPT	arxiv	1	39797
40784	In this work, we propose a scheme for continuo...	chatGPT	arxiv	1	40784
58487	Eeva-Kaarina Aronen (born July 6, 1961) is a F...	chatGPT	wikipedia	1	58487
...	...	...	...	...	...
14116	The X.28 was the first British jet aircraft to...	bloomz	wikipedia	4	14116
20596	The Crips is an alliance of street gangs which...	human	wikipedia	0	20596
54832	As with any illness, the more time you can gi...	human	wikihow	0	54832
13223	Nimbarkas are followers of the Nimbārka school...	bloomz	wikipedia	4	13223
67579	How to Tip Tips are passed along to service pr...	dolly	wikihow	5	67579

14206 rows × 5 columns

Figure 2-4: This is data validation

Figure 5-2: The results of the evaluation data in the training process (1 Epoch and50 =max_step And 17 minutes to compare with the second question that has adapter)

```
human: {'precision': 0.5, 'recall': 0.004, 'f1-score': 0.007936507936507936, 'support': 500}
chatGPT: {'precision': 0.3747474747474748, 'recall': 0.742, 'f1-score': 0.4979865771812081, 'support': 500}
cohere: {'precision': 0.20321931589537223, 'recall': 0.202, 'f1-score': 0.20260782347041123, 'support': 500}
davinci: {'precision': 1.0, 'recall': 0.0, 'f1-score': 0.0, 'support': 500}
bloomz: {'precision': 0.46065259117082535, 'recall': 0.96, 'f1-score': 0.6225680933852141, 'support': 500}
dolly: {'precision': 0.42398286937901497, 'recall': 0.396, 'f1-score': 0.4095139607032058, 'support': 500}
accuracy: 0.384
macro avg: {'precision': 0.4937670418654479, 'recall': 0.38399999999999995, 'f1-score': 0.2901021604460912, 'support': 3000}
weighted avg: {'precision': 0.49376704186544795, 'recall': 0.384, 'f1-score': 0.29010216044609116, 'support': 3000}
```

Figure 2-6: The results of the data test

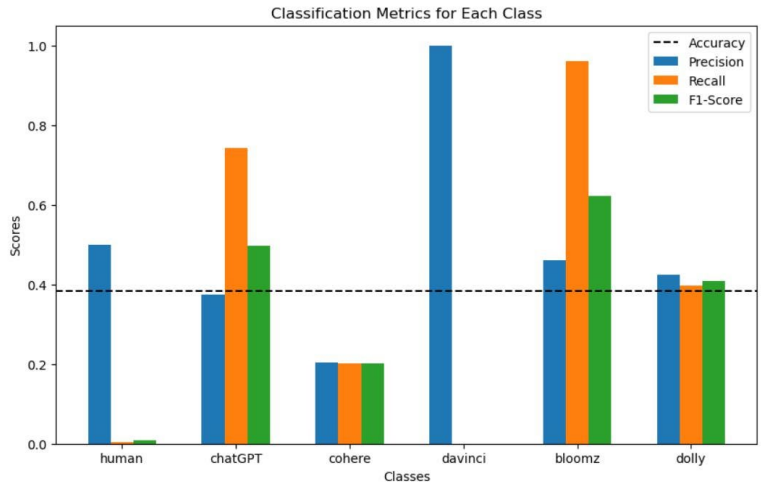


Figure 2-7: The results of the test data on the graph

We reached 38 percent accuracy for the epoch.

For 100=max_step We also have:

Figure 2-8: The results of the evaluation data in the training process (1 epoch and100 =max_stepand 24:10 minutes

```
human: {'precision': 0.6927374301675978, 'recall': 0.248, 'f1-score': 0.3652430044182622, 'support': 500}
chatGPT: {'precision': 0.3268398268398268, 'recall': 0.604, 'f1-score': 0.4241573033707865, 'support': 500}
cohere: {'precision': 0.2512363996043521, 'recall': 0.508, 'f1-score': 0.33620119126406356, 'support': 500}
davinci: {'precision': 0.0, 'recall': 0.0, 'f1-score': 0.0, 'support': 500}
bloomz: {'precision': 0.639686684073107, 'recall': 0.98, 'f1-score': 0.7740916271721958, 'support': 500}
dolly: {'precision': 0.3577981651376147, 'recall': 0.078, 'f1-score': 0.12807881773399016, 'support': 500}
accuracy: 0.403
macro avg: {'precision': 0.3780497509704164, 'recall': 0.40299999999999997, 'f1-score': 0.337961990659883, 'support': 3000}
weighted avg: {'precision': 0.37804975097041643, 'recall': 0.403, 'f1-score': 0.33796199065988297, 'support': 3000}
```

Figure 2-9: The results of the data test

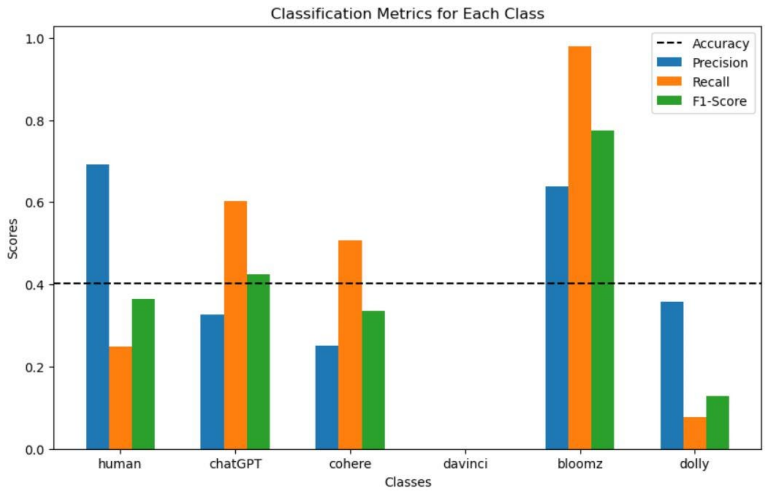


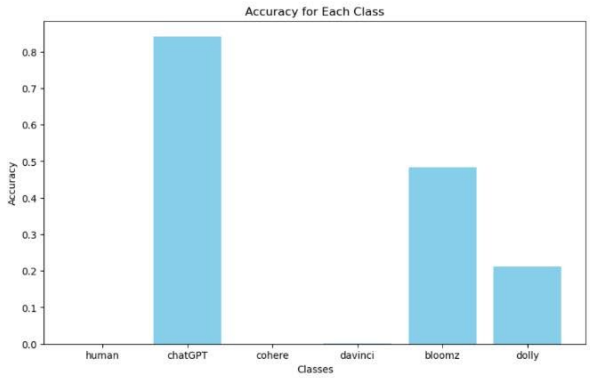
Figure 2-10: The results of the test data on the graph

We have reached from 38 to 40 for the epoch. As can be seen, the value can be max_step increased to reach higher accuracy.

It is necessary to mention that we have implemented this question in semi-supervised mode as well: First, we train labeled data, then we use it to train unlabeled data. Naturally, it takes a lot of time to learn to code from a technical point of view.

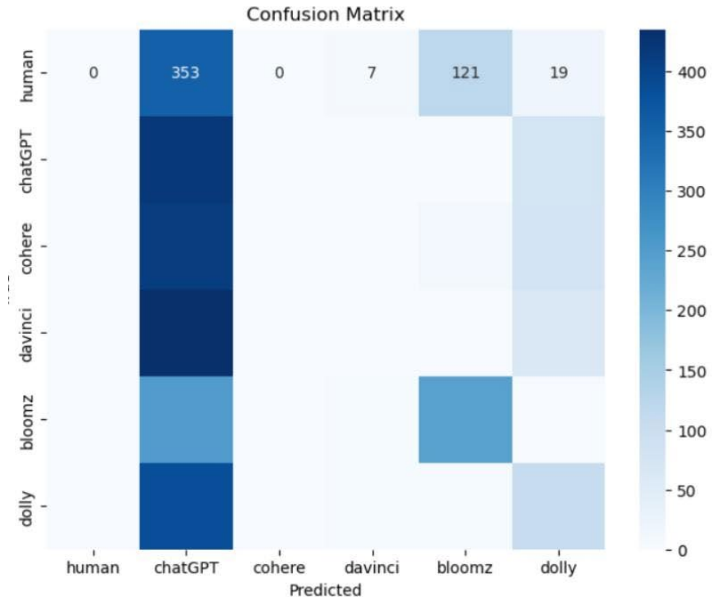
To this aim, self-training has been used. In this method, the model is first trained on labeled data. Then it uses its own predictions for unlabeled data to generate pseudo-labeled data. Finally, the model is trained on a mixture of labeled and pseudo-labeled data. This work can be repeated until any specific precision is achieved. The results for this method are as follows:

```
Accuracy: 0.2567
Accuracy for class human: 0.0000
Accuracy for class chatGPT: 0.8420
Accuracy for class cohere: 0.0000
Accuracy for class davinci: 0.0020
Accuracy for class bloomz: 0.4840
Accuracy for class dolly: 0.2120
```



Classification Report:

	precision	recall	f1-score	support
human	1.00	0.00	0.00	500
chatGPT	0.19	0.84	0.31	500
cohere	1.00	0.00	0.00	500
davinci	0.06	0.00	0.00	500
bloomz	0.64	0.48	0.55	500
dolly	0.31	0.21	0.25	500
accuracy			0.26	3000
macro avg	0.53	0.26	0.19	3000
weighted avg	0.53	0.26	0.19	3000



As can be seen, the accuracy is around 25% for 20% with labels and 80% without labels, which also took a long time for training (5 hours).

Chapter 3

second request

In this part We use adaptor to speed up the learning process.

3-1 Description of the method

The class Adaptor will be added as a layer to the pre-trained model BERT. In fact, it is a specific neural network module based on a known task that aims to fine-tune a pre-trained model is used without making many changes on it.

Feature Layers of adaptor are as follows:

- initial value (__init__):
  - input_size: The dimensions of the input layer (here, the dimensions of the hidden layer BERT)
  - output_size: The dimensions of the output layer
  - reduction_factor: Dimension reduction factor for middle layers adaptor
- adapter layers: Every layer adaptor consists of the following:
  - The line layer that reduces the input dimensions based on reduction_factor
  - ReLU

– Linear layer for output mapping

- Forward Pass

  – the method forward Determinant forward pass for adaptor It is on every layer adaptor It is repeated and applied to the input tensor.

  – skip connection By adding output adaptor It applies to the entry.

  – skip connection It will be necessary to save the original input information

How to use: adapter

It is enough to model a sample of this class BERT Let's add. The number of layers adaptor It should be equal to the number of encoder layers BERT be layers adaptor With the aim of reducing the trainable parameters of the model in the network, they are used to make the training process faster and more efficient and to improve the generalizability. A number of advantages adaptor They are in this order:

– Improving the efficiency of learnable parameters

– Efficiency of calculations

– Increasing the speed of learning

– Improving generalizability

– Transfer learningScalable

– Saving the knowledge of the pre-trained model

3-2 Results

In this part of the structure of the first question adaptor We will benefit. The following results are obtained:

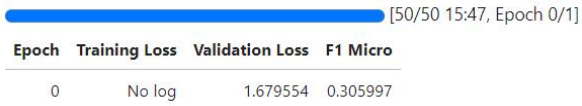


Figure 3-1: The results of the evaluation data in the learning process (1) Epoch and50 =max_stepand 15:47 minutes to

compare with the question

```
human: {'precision': 0.9, 'recall': 0.036, 'f1-score': 0.06923076923076922, 'support': 500}
chatGPT: {'precision': 0.32251235003528583, 'recall': 0.914, 'f1-score': 0.47678664580073027, 'support': 500}
cohere: {'precision': 0.371900826446281, 'recall': 0.54, 'f1-score': 0.4404567699836868, 'support': 500}
davinci: {'precision': 1.0, 'recall': 0.0, 'f1-score': 0.0, 'support': 500}
bloomz: {'precision': 0.7814207650273224, 'recall': 0.286, 'f1-score': 0.4187408491947291, 'support': 500}
dolly: {'precision': 0.3119266055045872, 'recall': 0.408, 'f1-score': 0.35355285961871746, 'support': 500}
accuracy: 0.364
macro avg: {'precision': 0.6146267578355794, 'recall': 0.36400000000000005, 'f1-score': 0.29312798230477216, 'support': 3000}
weighted avg: {'precision': 0.6146267578355794, 'recall': 0.364, 'f1-score': 0.29312798230477216, 'support': 3000}
```

Figure 3-2: The results of the test data

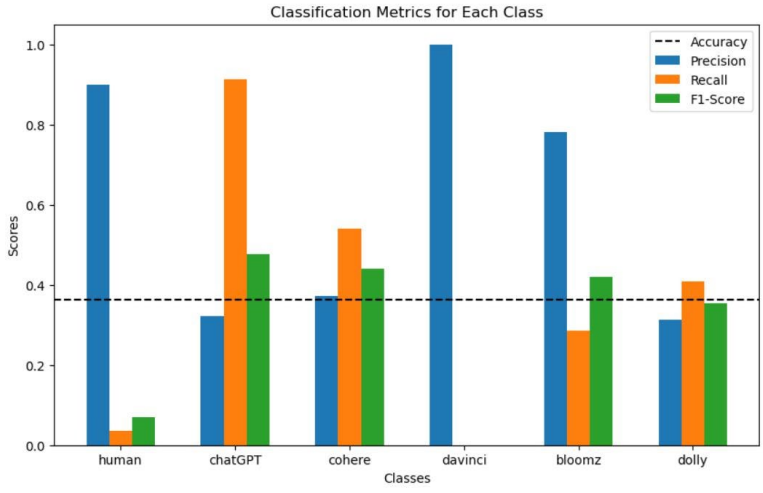


Figure 3-3: The results of the test data on the graph

As you can see, we went from 17 minutes to 15:47 minutes, which can be done by adjusting the invoice. reduction_factor in adapter It reduced the time more. Also, we reached 36 percent accuracy, which is a 2 percent decrease compared to the first question, which is due to the structure adapter which reduces the parameters of the model

Chapter 4

The third request

In this section We will pay attention to implementation of GAN-BERT.

4-1 Description of the method

Construction of the night generator GAN-BERTIt can be done with two methods which are listed respectively1-2)1G(And1-3)2G (has been shown. The details of each method are explained in the introduction chapter. However, in the continuation of the code implementation details2GCompared to1GIt will be explained: it is enough to change the following items in the code:

Class Generator
Bag_of_Words
Instanse of Generator (g_2)

- in class *G_2*, from y and in Construction model BERTin the name of 'bert-base-uncased We use the exit layer from y We use the risk layer to produce the label produced by the manufacturer. in stage forward function, which Bag of words receives in his login, an example of coincidence
  He took it and after Tokenize to do by Tokenizer yourself BertTokenizerModel it
  BertLet's go through the data and the line.

- For the bag of words, we use the training file without tags, which has a comprehensive section of words, and each word of it with Tokenizer Introduced in the previous section, Tokenize We will add it to the list bag_of_words We will add the final list as an exampleg2from the classGenerator2 We give
- At the end of Instantiate made from his own class and instead of it we use in the learning circle g1.

The analysis of two known generators:

The generator model *G_2* Compared to the original model *G_1* It has several advantages and disadvantages:

Advantages:

1. Using the modelBERTTrained:2*G*He uses his modelBERTHe has been trained for Ten of text understanding and language reproduction abilitiesBERT generator night and is safe with it. benefit This can be fake samples In terms of meaning, it will be more meaningful and relevant.

2. Actual output: using a set of words from the training samples. To produce a sample Coincidence, purpose2*G*It is the production of more real and consistent fake samples that represent the training data. This can potentially improve the quality of produced samples.

3. Simplified noise input:2*G*like1*G*It does not depend on the noise vector. Instead, it uses frequency-based random sampling from the bag of words, which should result in a more structured and meaningful input for the generator.

Disadvantages:

1. Complexity: Model2*G*Add more complexity to the mix. This leads to an increase in the needs of calculations. modelBERTfrom prior knowledge and training time compared to simpler architecture

1*G*be

2. Depending on the model of pre-training: reliance2*G*To Y modelBERTHe has seen it from his training

It depends on the quality and correlation of the data before the training. If the modelBERTThe training received is not suitable for the job or the specific field, but it is effective.2*G*limit

More explanation of simplicity and accuracy:

- Model2GThe whole model of GAN-BERTdue to the additional complexity of integrating a modelBERTprevent

Using understanding of the textBERTAnd education is not necessarily simpler However, it has the potential to increase the accuracy of the produced samples with

export production

Give more real improvements.

- Accuracy of the model2GIn the scenarios where this work requires a deep understanding of the language and time, it is better, because the modelBERTFrom pre-learning, you can describe the complex relationships of the language. However, the actual improvement in the accuracy of the specific task, the data set, and the quality of the model.BERTIt depends on the training.

In summary, the model2GWith the integration of a modelBERTIt introduces a more complex process than pre-trained, which can potentially lead to more relevant and realistic forgery patterns. However, this is more complicated and depends on the quality of the pre-trained model. The effect on the accuracy and simplicity of the model should be evaluated experimentally in the context of specific tasks and data sets.

4**-**2 Results

- First, the results for the model1GAnd considering 20 percent of the data is labeled as data and We check 08 percent without a label:

```
  Average training loss generetor: 0.676
  Average training loss discriminator: 1.942
  Training epcoh took: 0:18:25

Running Test...
  Accuracy: 0.437
  F1 Score: 0.393
  Test Loss: 1.857
  Test took: 0:00:08
```

Figure 4-1: Epoch 1

```
  Average training loss generetor: 0.711
  Average training loss discriminator: 1.131
  Training epcoh took: 0:23:00

Running Test...
  Accuracy: 0.405
  F1 Score: 0.370
  Test Loss: 2.203
  Test took: 0:00:13
```

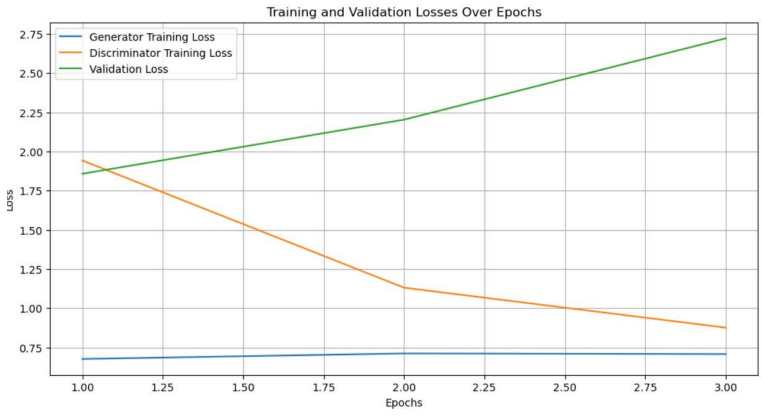
Figure 4-2: Epoch 2

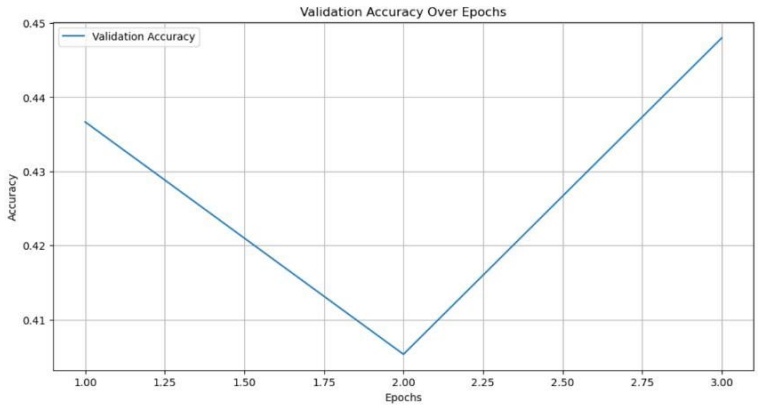
```
  Average training loss generetor: 0.707
  Average training loss discriminator: 0.876
  Training epcoh took: 0:24:06

Running Test...
  Accuracy: 0.448
  F1 Score: 0.398
  Test Loss: 2.721
  Test took: 0:00:08
```

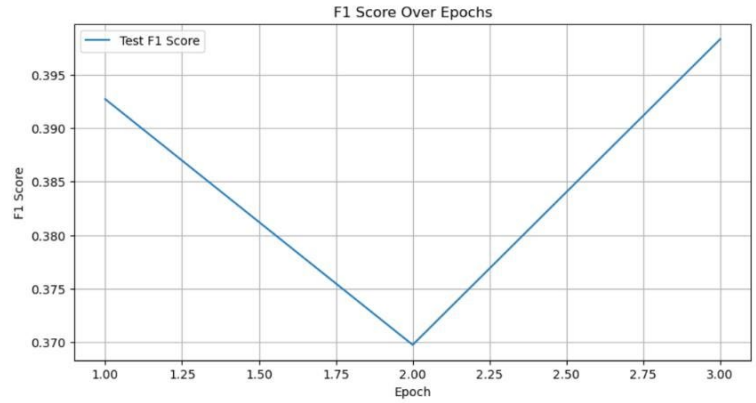
Figure 4-3: Epoch 3

```
{'epoch': 1, 'Training Loss generator': 0.6763042037461972, 'Training Loss discriminator': 1.9423223393367934, 'Valid. Loss': 1.8574364185333252, 'Valid.
Accur.': 0.43666666666666665, 'Training Time': '0:18:25', 'Test Time': '0:00:08'}
{'epoch': 2, 'Training Loss generator': 0.7114604089084688, 'Training Loss discriminator': 1.1309138255434352, 'Valid. Loss': 2.202589988708496, 'Valid.
Accur.': 0.4053333333333333, 'Training Time': '0:23:00', 'Test Time': '0:00:13'}
{'epoch': 3, 'Training Loss generator': 0.707479796431086, 'Training Loss discriminator': 0.8759377081920435, 'Valid. Loss': 2.7212116718292236, 'Valid.
Accur.': 0.448, 'Training Time': '0:24:06', 'Test Time': '0:00:08'}

Training complete!
Total training took 1:06:01 (h:mm:ss)
```

Figure 4-4: Summary of results







- model1G40 percent with labels, 60 percent without labels:

  By increasing the label data, the result will be better, compared to the previous case, you can see that the training fluctuation has been removed, and if the previous case needed 10 epochs, this case needs less.

```
  Average training loss generetor: 0.666
  Average training loss discriminator: 2.084
  Training epcoh took: 0:15:49

Running Test...
  Accuracy: 0.455
  F1 Score: 0.414
  Test Loss: 1.522
  Test took: 0:00:10
```

Figure 4-5: Epoch 1

```
  Average training loss genereтor: 0.712
  Average training loss discriminator: 1.372
  Training epcoh took: 0:16:31

Running Test...
  Accuracy: 0.467
  F1 Score: 0.456
  Test Loss: 1.625
  Test took: 0:00:09
```

Chapter 4-6: Epoch 2

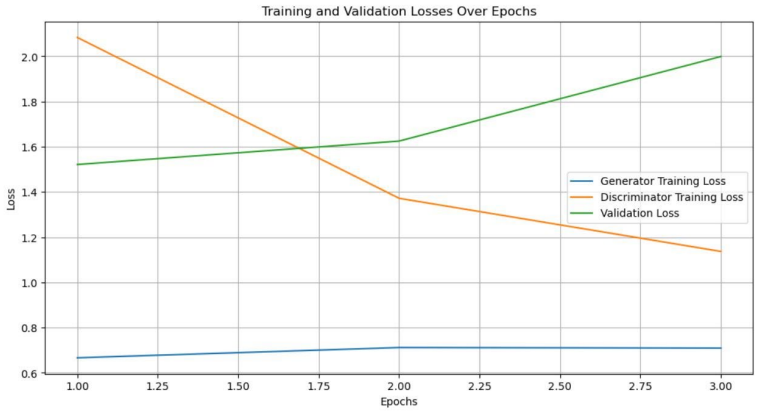
```
  Average training loss genereтor: 0.709
  Average training loss discriminator: 1.137
  Training epcoh took: 0:17:34

Running Test...
  Accuracy: 0.471
  F1 Score: 0.440
  Test Loss: 1.999
  Test took: 0:00:13
```

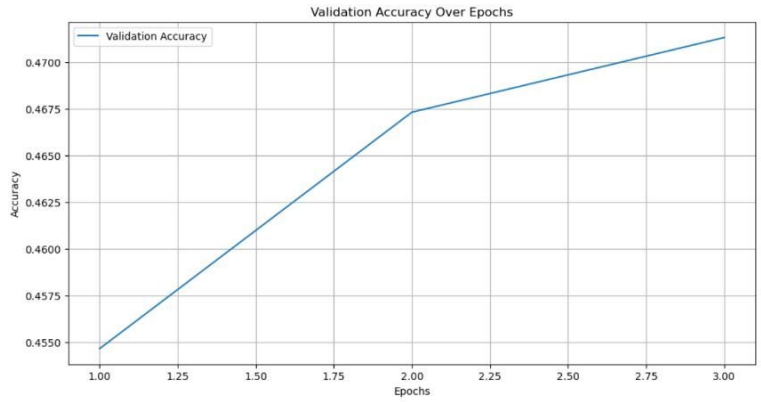
Chapter 4-7: Epoch 3

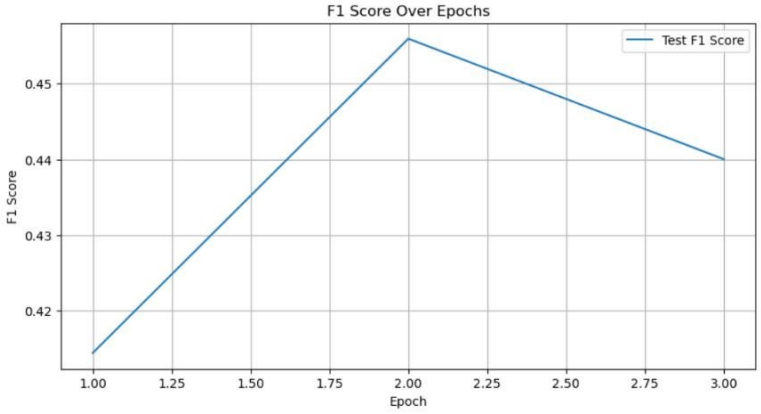
{'epoch': 1, 'Training Loss generator': 0.6664814874932573, 'Training Loss discriminator': 2.0835476065541174, 'Valid. Loss': 1.5215482711791992, 'Valid. Accur.': 0.45466666666666666, 'Training Time': '0:15:49', 'Test Time': '0:00:10'}
{'epoch': 2, 'Training Loss generator': 0.7118143864580103, 'Training Loss discriminator': 1.372109925156241, 'Valid. Loss': 1.6251945495605469, 'Valid. Accur.': 0.4673333333333333, 'Training Time': '0:16:31', 'Test Time': '0:00:09'}
{'epoch': 3, 'Training Loss generator': 0.7094472833044894, 'Training Loss discriminator': 1.137035841823698, 'Valid. Loss': 1.999358057975769, 'Valid. Accur.': 0.4713333333333333, 'Training Time': '0:17:34', 'Test Time': '0:00:13'}

```
Training complete!
Total training took 0:50:26 (h:mm:ss)
```

Figure 4-8: Summary of results







Analysis of architectural results *G_1*:

- LossEducation for Shabbat:

6665.0 : 1*Epoch*

7118.0 : 2*Epoch*

7094.0 : 3*Epoch*

LossIt is a standard generator that shows lower Deh is able to produce real samples. values are usually better, because it shows Let it be productive in creating real outputs It is improving. A slight increase in the losses of the generator night from Epoch 1 to Epoch 2 and then a relatively stable value in Epoch 3 may indicate a variable value, but it does not necessarily indicate a problem.

- LossTraining for night cleaner:

0835.2:1*Epoch*

3721.1:2*Epoch*

1370.1:3*Epoch*

LossThe discriminator measures the degree of differentiation between real and produced samples. Similar to LossFor generators, lower values are generally better. Significant reduction inLossThe discriminator from Epoch 1 to Epoch 3 shows that the generator has improved in deceiving the discriminator.

- LossCreditor and:accuracy

4547.0, *Accuracy*5215.1*Loss*:1*Epoch*

4673.0, *Accuracy*6252.1*Loss*:2*Epoch*

4713.0, *Accuracy*9994.1*Loss*:3*Epoch*

LossThe reliability indicator shows how well the model generalizes to the unseen data. increase LossThe credit from Epoch 1 to Epoch 3 is too much to show. The accuracy is relatively stable, but not very high.

- Study and test time:

*seconds*49*minutes*15:1*Epoch*

*seconds*31*minutes*16:2*Epoch*

*seconds*34*minutes*17:3*Epoch*

The training time is increasing with each epoch, which can be due to the learning of more complex model sheets or the large size of the set.

- model2G20 percent with labels, 80 percent without labels:

```
  Average training loss generetor: 0.722
  Average training loss discriminator: 1.655
  Training epcoh took: 0:19:29

Running Test...
  Accuracy: 0.441
  F1 Score: 0.398
  Test Loss: 1.918
  Test took: 0:00:08
```

Chapter 4-9: Epoch 1

```
  Average training loss generetor: 0.712
  Average training loss discriminator: 1.051
  Training epcoh took: 0:26:17

Running Test...
  Accuracy: 0.436
  F1 Score: 0.387
  Test Loss: 2.312
  Test took: 0:00:13
```

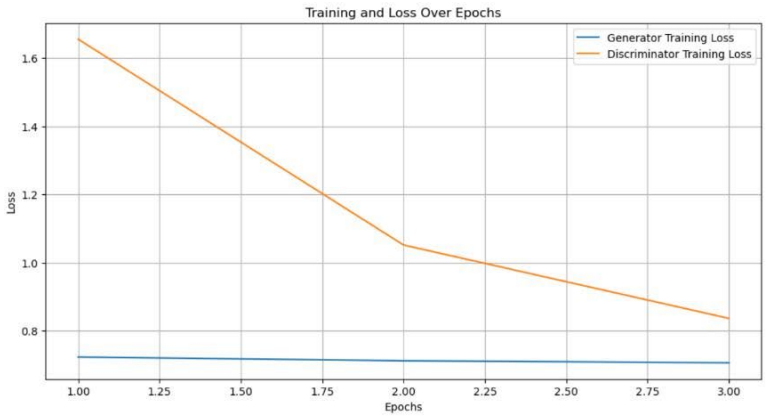
Chapter 4-10: Epoch 2

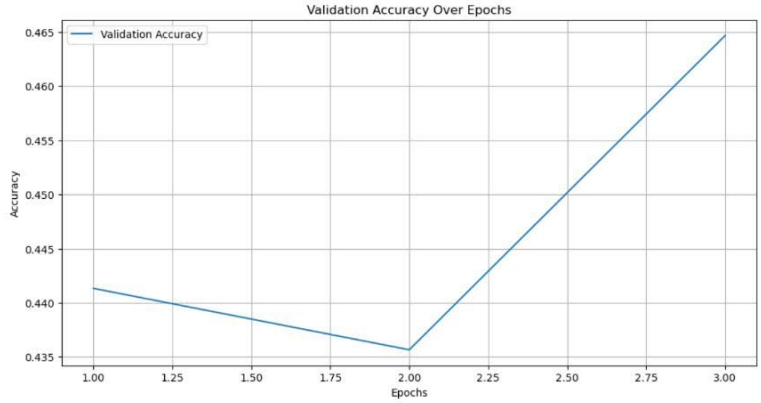
```
  Average training loss generetor: 0.706
  Average training loss discriminator: 0.836
  Training epcoh took: 0:22:20

Running Test...
  Accuracy: 0.465
  F1 Score: 0.426
  Test Loss: 2.879
  Test took: 0:00:11
```

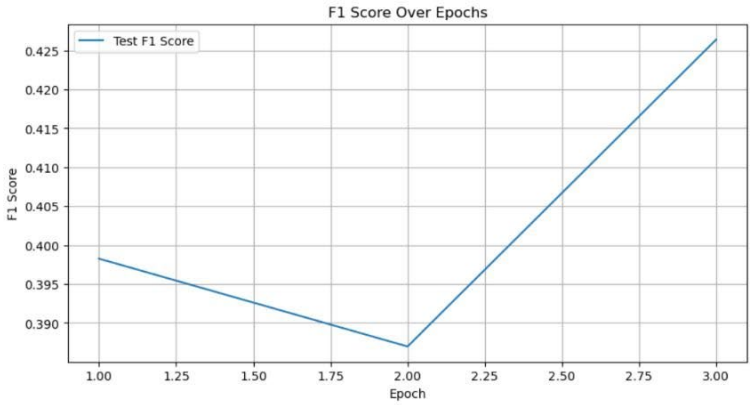
4-11: Epoch 3

```
{'epoch': 1, 'Training Loss generator': 0.7224929826529892, 'Training Loss discriminator': 1.6550412992009886, 'Valid. Accur.': 0.44133333333333336, 'Tra
ining Time': '0:19:29', 'Test Time': '0:00:08'}
{'epoch': 2, 'Training Loss generator': 0.7116552988240669, 'Training Loss discriminator': 1.0511942154115386, 'Valid. Accur.': 0.43566666666666665, 'Tra
ining Time': '0:26:17', 'Test Time': '0:00:13'}
{'epoch': 3, 'Training Loss generator': 0.7057131770942304, 'Training Loss discriminator': 0.8359519980243735, 'Valid. Accur.': 0.4646666666666667, 'Trai
ning Time': '0:22:20', 'Test Time': '0:00:11'}

Training complete!
Total training took 1:08:37 (h:mm:ss)
```

Figure 4-12: Summary of results







Be careful1Gequal to0.436It was and in2G0.465became

- model2G50% with label, 50% without label: with the

  increase of labelled data, the result will be better:

```
  Average training loss generetor: 0.731
  Average training loss discriminator: 1.754
  Training epcoh took: 0:22:24

Running Test...
  Accuracy: 0.464
  F1 Score: 0.414
  Test Loss: 1.524
  Test took: 0:00:07
```

4-13: Epoch 1

```
  Average training loss generetor: 0.720
  Average training loss discriminator: 1.276
  Training epcoh took: 0:16:59

Running Test...
  Accuracy: 0.506
  F1 Score: 0.487
  Test Loss: 1.689
  Test took: 0:00:11
```

Chapter 4-14: Epoch 2

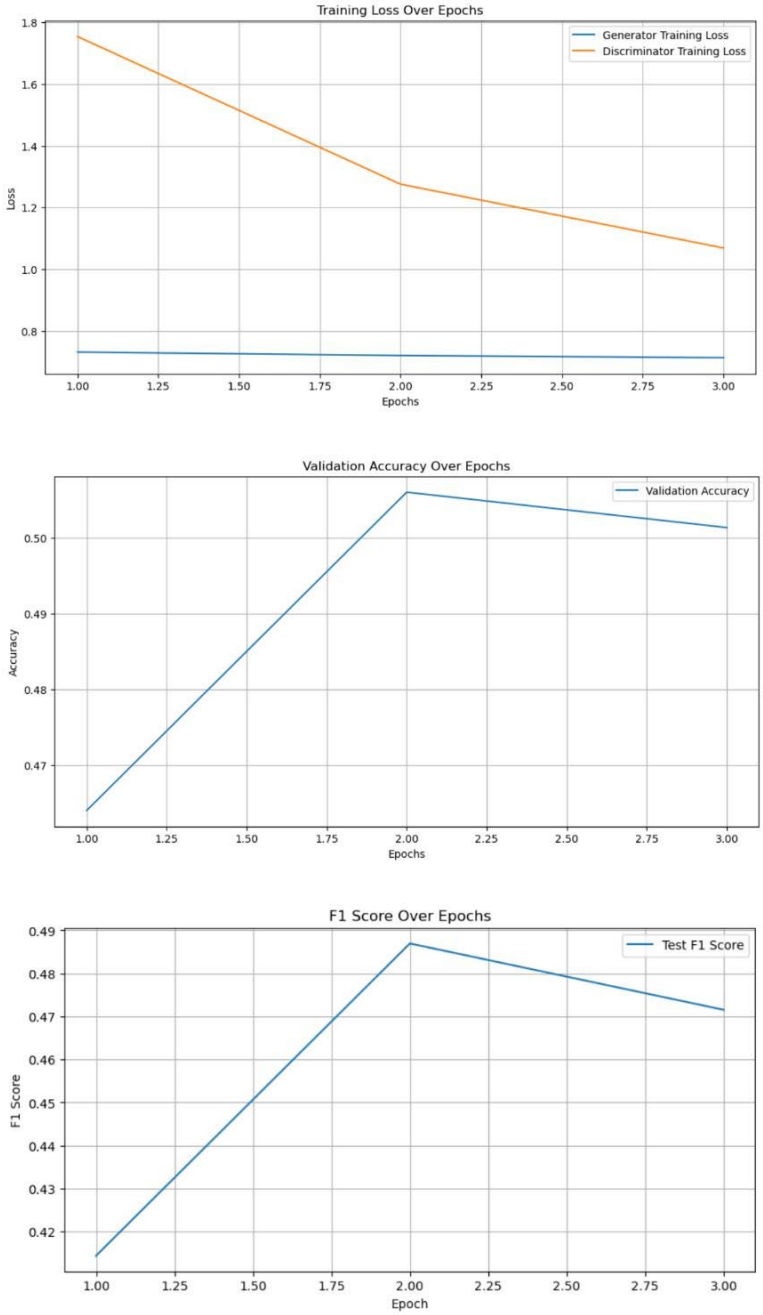
```
  Average training loss generetor: 0.713
  Average training loss discriminator: 1.069
  Training epcoh took: 0:17:27

Running Test...
  Accuracy: 0.501
  F1 Score: 0.472
  Test Loss: 2.024
  Test took: 0:00:10
```

Chapter 4-15: Epoch 3

```
{'epoch': 1, 'Training Loss generator': 0.7310682469391608, 'Training Loss discriminator': 1.7543531511280988, 'Valid. Accur.': 0.464, 'Training Time':
'0:22:24', 'Test Time': '0:00:07'}
{'epoch': 2, 'Training Loss generator': 0.7199606829935365, 'Training Loss discriminator': 1.2755506262048946, 'Valid. Accur.': 0.506, 'Training Time':
'0:16:59', 'Test Time': '0:00:11'}
{'epoch': 3, 'Training Loss generator': 0.7128319162237752, 'Training Loss discriminator': 1.0686820583300547, 'Valid. Accur.': 0.5013333333333333, 'Trai
ning Time': '0:17:27', 'Test Time': '0:00:10'}

Training complete!
Total training took 0:57:17 (h:mm:ss)
```

Figure 4-16: Summary of results

Be careful1G0.51(4 epochs, 1 hour and 14 minutes) and in2G0.501It ended with (3 epochs of 57 minutes).



As seen in the results set for2Gbetter than1GIts reasons were explained in the previous subsection.

Chapter 5

The fourth request (scoring)

To improve in this chapterGAN-BERTwith architecture1GWe will pay.

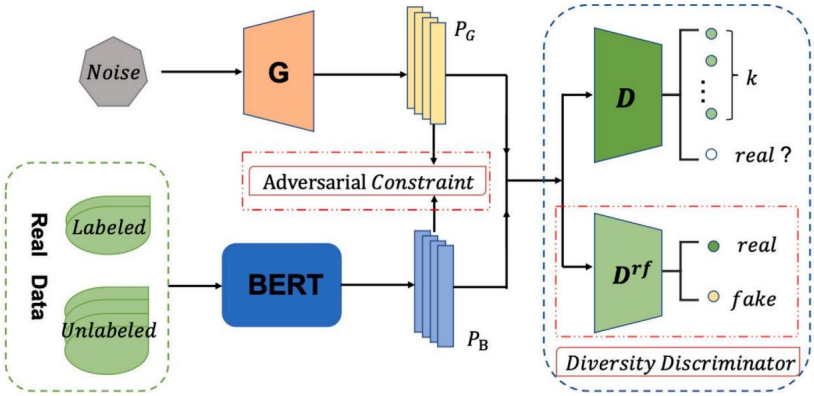
5-1 Description of the method

To improve the model, two sets of measures have been taken:

1.Improvement of the generator using the following changes:

- Adjust the amountdropout
- Add a layerBatch NormalizationTo the hidden layers

2. Using two separate cleaning methods, one to identify the real or fake samples and the other As a stratum (5-1)



Question 5-1:GAN-BERTwith twodiscriminator

References

[1] Kenton, Jacob Devlin Ming-Wei Chang and Toutanova, Lee Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. in Proceedings of naacL-HLT, vol. 1, p. 2, 2019.

[2] Croce, Danilo, Castellucci, Giuseppe, and Basili, Roberto. Gan-bert: Genera□tive adversarial learning for robust text classification with a bunch of labeled examples. 2020