Evaluation of Artificial Intelligence Text Detection Models

Dana Goh Siew Yuen[1], Tan Wei Peng, Andrew[2],
Nanyang Girls' High School, 2 Linden Dr, Singapore 288683
Defence Science and Technology Agency, 1 Depot Rd, Singapore 109679

1. Abstract

The present study evaluates existing open-source offline Artificial Intelligence (AI) text detector models. This was done by selecting models that met the criteria and running them through a self-curated dataset with English text. The dataset consisted of an equal number of AI-generated and human-generated text which spanned over a large variety of formal and informal topics. Models that performed well were then run through an expanded dataset including Chinese and Malay text to determine the accuracy of the detectors. The models were judged based on their accuracy, precision, recall, F1 score, and confusion matrix derived from the outcome of the tests. The results showed that the Binoculars model was the most proficient detector across all languages. Many models were outdated and did not perform well on the newer versions of GPT. For those that did, their accuracy decreased as different languages were added to the dataset. This shows that existing models must be trained regularly with newer versions of GPT and multiple languages to keep up with current trends and be viable in today's society.

2. Introduction

In this digital era, technology is advancing at a rapid pace. Currently, a key focus in technology is the development of AI and Large Language Models (LLMs). Due to the rise of LLMs like ChatGPT and Google Gemini, these products have become accessible to the general public. While this has allowed for increased efficiency, automating and enhancing numerous tasks involving natural language processing amongst multiple advantages, many ethical concerns have also been raised. For example, because students have easy access to LLMs nowadays, they may use chatbots to write graded reports. Without a proper AI text detector, it is difficult for their teachers to determine the authenticity of their reports. Hence, quickly developing an accurate AI text detector in Singapore is crucial to prevent such events.

Very few existing models have high accuracy rates in detecting Malay and Chinese text, the two most commonly used languages in Singapore apart from English. The ability to accurately detect AI-generated text in these languages is crucial, as it is not just about detecting AI content, but doing so in a culturally and linguistically sensitive manner. Developing a model that addresses the specific needs of our multilingual society will require a lot of time, money, and resources. Hence, to aid with the development of such a model, this project aims to find the best open-sourced detector to fine-tune detection models in the local context. Currently, most models with high accuracy rates in detecting AI-generated text are closed-sourced. This project evaluates the open-source landscape and picks the best generative text detection model which can be compared to paid software.

3. Materials and Methods

3.1. Sources for generative text detection models

The sites where models were sourced include HuggingFace[2], GitHub[3], and online research papers. Suitable models were deemed as those with Application Programming Interface (API) access, good ratings and many downloads, allowed training and supported multiple languages. The models were then collated in a spreadsheet with their strengths and limitations. Those that seemed most suitable were selected for testing. The models selected were AraGPT2 Detector[4], Roberta Base[5], Bert Lli[6], AIGC Detector (MPU)[7], MayZhou[8], Nintw923 Chatgpt Detector LLI HC3[9], SuperAnnotate[10], MayZhou e5 small lora ai generated detector[8], XLM-RoBERTa (base) fine-tuned on HC3[11], and Binoculars[12].

3.2. Collection of data for self-curated dataset

The dataset consists of an equal number of AI-generated text and human-generated text. There were 202 text samples in total, with an equal number of formal and informal text samples. 60% of the dataset was in English, 20% of the dataset was in Chinese, and the last 20% was in Malay.

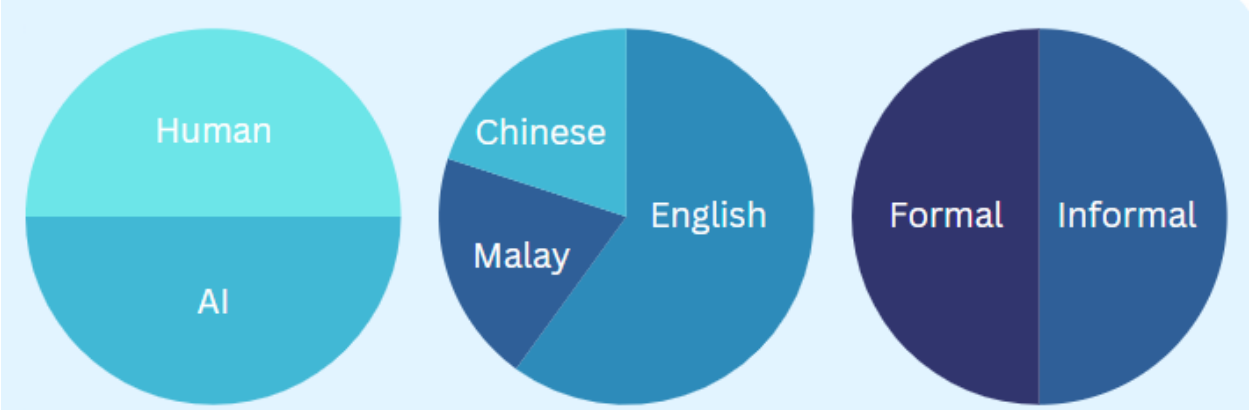
Dataset 1 comprises 100 English text samples, with an equal number of AI-generated and human-generated text, and an equal number of formal and informal text samples. Dataset 2 comprises 120 English text samples, 42 Chinese text samples, and 40 Malay text samples, with an equal number of AI-generated and human-generated text, and an equal number of formal and informal text samples. The breakdown of the data can be found in Graphs 3.2.2, 3.2.3, and 3.2.4.

All the AI-generated text samples were generated through ChatGPT 4.0. Prompts given to ChatGPT spanned a large range of topics, varying the length, tone, and format of the prompt each time. This ensured that the text samples were diverse enough to properly test the detectors. Human-generated text samples were taken from datasets used by previous researchers on Kaggle[13] and Huggingface[14], as well as blogs and news articles written before 2018 as generative AI was not widely used.

Figure 3.2.1 shows the first 10 rows of Dataset 2.

	A	B	C	D	E	F	G
	Table2						
1	text	generated	language	source	formality	topic	remarks
2	Cars. Cars have been around since they became f	human	english	kaggle	formal	cars	
3	My Pet Cat: The Queen of the House	ai	english	chatgpt	informal	cats	
4	I got my boy Joe from a shelter right at the beginn	human	english	reddit	informal	cats	
5	Okay, imagine when you tap a drum, and it makes	ai	english	chatgpt	informal	vibroacoustics	
6	Chocolate cupcakes filled with caramel, salted car	human	english	reddit	informal	cupcakes	
7	Why Sour Cream and Onion Chips Is the Superior I	ai	english	chatgpt	informal	chips	
8	In string musical instruments, the sound is radiate	human	english	huggingface	formal	virboacoustics	
9	Why Sour Cream an Onion Chips r the Best, Perioc	ai	english	chatgpt	informal	chips	super informal
10	**The Art of Flirting is Dead, Change My Mind**	ai	english	chatgpt	informal	flirting	short

Figure 3.2.1



Graph 3.2.2 Graph 3.2.3 Graph 3.2.4

3.3. Testing of the models using the datasets

Kaggle was used to load the Binoculars model, while Google Colab was used to load the other models. This is because Binoculars required a higher amount of GPU memory to load two LLM models to perform its scoring which was not available in the free version of Google Colab. The code remained almost the same.

All the suitable models were run through Dataset 1 to see how each model performs in the initial phase. To import the open-source models to Google Colab, transformers, datasets, and tokenizers were installed. The model usage code was then referenced from the individual model documentation and executed on Google Colab.

```
# IMPORTING MODEL ------------------------------------------------------------

!pip install transformers
!pip install datasets
!pip install tokenizers

# Load model directly
from transformers import AutoTokenizer, AutoModelForSequenceClassification

tokenizer = AutoTokenizer.from_pretrained("mrm8488/xlm-roberta-base-finetuned-HC3-mix")
model = AutoModelForSequenceClassification.from_pretrained("mrm8488/xlm-roberta-base-finetuned-HC3-mix")

# Use a pipeline as a high-level helper
from transformers import pipeline

pipe = pipeline("text-classification", model="mrm8488/xlm-roberta-base-finetuned-HC3-mix")
```

Figure 3.3.1

```
# GETTING DATA FROM GOOGLE SHEETS ------------------------------------------------
from google.colab import auth
auth.authenticate_user()

import gspread
from google.auth import default
creds, _ = default()

gc = gspread.authorize(creds)

dataset = gc.open('YDSP Dataset').sheet1

values = dataset.get_all_values()
```

Figure 3.3.2

Each model was run to predict the entire 100 rows of data in Dataset 1. A score of 1 was returned if the model detected AI-generated text and a score of 0 was returned if it detected human text, as shown in Figure 3.3.3. The individual prediction results were then compared to the actual labels to see if it was predicted accurately.

```
# GET ORIGINAL RESULTS ------------------------------------------------------------
original_results = []

for i in range(100):
  r = values[i+1][1]

  if r == "human":
    original_results.append(0)

  else:
    original_results.append(1)

print(original_results)
```

Figure 3.3.3

```
# GET RESULTS ---------------------------------------------------------------------
model_results = []

for i in range(100):
  text = values[i+1][0]

  text_length = len(text)
  if text_length > 1500:
    text = text[0:1500]

  result = pipe(text)
  print(i+1, result)
  pos_or_neg = result[0].get("label")

  if pos_or_neg == "Human":     #human
    model_results.append(0)

  else:     #ai
    model_results.append(1)

print(model_results)
```

Figure 3.3.4

The accuracy, confusion matrix, precision, recall, and F1 score were then calculated using the sklearn.metrics module, as shown in Figure 3.3.5.

```
# accuracy score
from sklearn.metrics import accuracy_score
print("accuracy score:", accuracy_score(original_results, model_results) * 100, "%")

# confusion matrix (tn, fp, fn, tp)
from sklearn.metrics import confusion_matrix
matrix = confusion_matrix(original_results, model_results)
tn = matrix [0][0]
fp = matrix [0][1]
fn = matrix [1][0]
tp = matrix [1][1]
print(f"confusion matrix: tn {tn}, fp {fp}, fn {fn}, tp {tp}")

# precision
from sklearn.metrics import precision_score
print("precision score:", precision_score(original_results, model_results, average=None))

#recall
from sklearn.metrics import recall_score
print("recall score:", recall_score(original_results, model_results, average=None))

# f1 score
from sklearn.metrics import f1_score
print("f1 score:", f1_score(original_results, model_results, average=None))
```

Figure 3.3.5

The results were then stored in Google Sheets and the top four models with the highest accuracy were selected to be run through Dataset 2 instead of Dataset 1. Dataset 2 included Chinese and Malay text on top of English text. This was done because in Singapore, Chinese and Malay are two commonly used languages on top of English, hence a model that can accurately detect all three languages would be an ideal model to sample code from. Running the top four models through Dataset 2 further tests the effectiveness of these models.

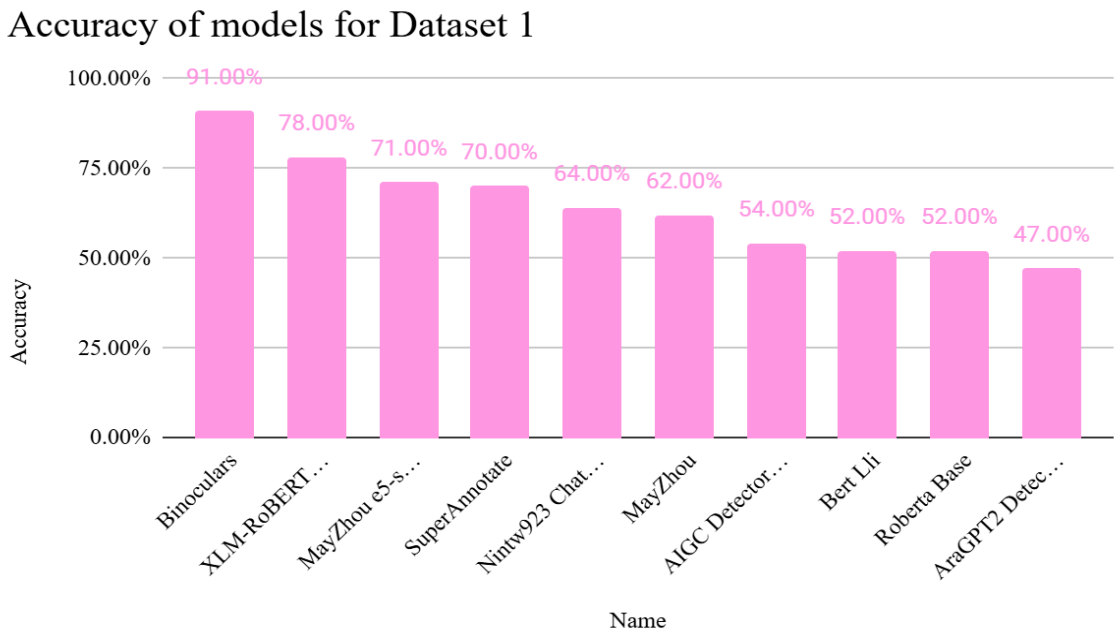
4. Results and analysis

4.1. Test with Dataset 1

The results of all the models that were tested with Dataset 1 are shown below in Table 4.1.1 and Graph 4.1.2.

Name	Accuracy	Confusion Matrix
Roberta Base	52.00%	tn 26, fp 24, fn 24, tp 26
MayZhou	62.00%	tn 27, fp 23, fn 15, tp 35
Bert Lli	52.00%	tn 26, fp 24, fn 24, tp 26
AraGPT2 Detector	47.00%	tn 2, fp 48, fn 5, tp 45
Nintw923 Chatgpt Detector LLI HC3	64.00%	tn 34, fp 16, fn 20, tp 30
AIGC Detector (MPU)	54.00%	tn 49, fp 1, fn 45, tp 5
MayZhou e5-small-lora-ai-generated-detector	71.00%	tn 22, fp 28, fn 1, tp 49
XLM-RoBERTa (base) fine-tuned on HC3	78.00%	tn 36, fp 14, fn 8, tp 42
Binoculars	91.00%	tn 49, fp 1, fn 8, tp 42
	70.00%	tn 48, fp 2, fn 28, tp 22
SuperAnnotate	70.00%	tn 33, fp 17, fn 13, tp 37

Table 4.1.1



Graph 4.1.2

Binoculars had the highest accuracy rate with Dataset 1 at 91.00%, with a false positive of only 1. However, 60% of the detectors had an accuracy of below 70.00%. This shows that there are likely many open-source detectors that are inaccurate at detecting text. For example, AraGPT2 had an accuracy rate of 47.00%, which is likely because it was trained on GPT 2. However, while sourcing, there were very few models that were trained on the current GPT 4 that powers popular AI chatbots. This suggests that a stronger AI detector needs to be trained with GPT 4 to be viable in today's society. Based on the results after testing with Dataset 1, MayZhou e5,

XLM-RoBERTa (base), Binoculars, and SuperAnnotate were selected to undergo testing with Dataset 2.

4.2. Test with Dataset 2

For the second test with Dataset 2, the results for each text sample were listed for comparison against the respective models. A sample of the first 25 results are shown in Table 4.2.1.

	A	B	C	D	E	F	G	H
1	content	language	category	truth	MayZhou e5	XLM-RoBERTa	Binoculars	SuperAnnotate
2	Cars. Cars have been around since they	english	formal	human	ai	human	human	human
3	My Pet Cat: The Queen of the House	english	informal	ai	ai	ai	ai	ai
4	I got my boy Joe from a shelter right at th	english	informal	human	human	human	human	human
5	Okay, imagine when you tap a drum, and	english	informal	ai	ai	ai	ai	ai
6	Chocolate cupcakes filled with caramel, s	english	informal	human	ai	human	human	ai
7	Why Sour Cream and Onion Chips Is the	english	informal	ai	ai	ai	human	human
8	In string musical instruments, the sound is	english	formal	human	ai	human	human	human
9	Why Sour Cream an Onion Chips r the Be	english	informal	ai	human	human	human	human
10	**The Art of Flirting is Dead, Change My I	english	informal	ai	ai	human	human	human
11	A new approach to vacuum decay in quar	english	formal	human	ai	human	human	human
12	There are many effects of modern techno	english	formal	human	ai	human	human	human
13	I love Oreos! They are the best cookie ev	english	informal	ai	ai	ai	ai	ai
14	Eras Tour Front Row Experience: I'm Still	english	informal	ai	ai	human	ai	human
15	So, my son has had an incredible sense o	english	informal	human	ai	human	human	human
16	We reveal that classical light diffraction in	english	formal	human	ai	human	human	ai
17	Okay, listen up, because I NEED to vent.	english	informal	ai	ai	human	ai	human
18	Skiing: The Good, The Bad, and The Cold	english	informal	ai	ai	human	ai	human
19	In modern society, cars have become an	english	formal	ai	ai	ai	ai	ai
20	My sister is raising iPad kids	english	informal	human	ai	human	human	human
21	We describe a theory of finite sets, and in	english	formal	human	ai	human	human	ai
22	Every morning, I wake up to the sight of s	english	formal	human	ai	human	human	human
23	The Art of Flirting is Dead	english	informal	human	human	human	human	human
24	Red Velvet Cupcakes	english	formal	ai	ai	ai	ai	ai
25	The Impact of Technology on Education: A	english	formal	ai	ai	ai	ai	ai

Table 4.2.1

The accuracy of all the detectors decreased when using the expanded dataset including Chinese and Malay. The accuracy of SuperAnnotate decreased the least by 8.12%, from 70.00% down to 61.88%, while the accuracy of MayZhou decreased the most by 15.06%, from 71.00% down to 80.69%, as shown in Table 4.2.2. The accuracy, precision, recall, F1 score, and confusion matrix of the four models are shown in Table 4.2.3. MayZhou e5 had the highest tendency to guess that AI-generated the text, with a total number of 173 guesses. Binoculars had the highest tendency to guess that content was human-generated, with a total number of 139 guesses.

Accuracy	SuperAnnotate	MayZhou e5	XLM-RoBERTa	Binoculars
Dataset 1	70.00%	71.00%	78.00%	91.00%
Dataset 2	61.88%	55.94%	60.39%	80.69%
Difference	-8.12%	-15.06%	-17.61%	-10.31%

Table 4.2.2

content	MayZhou e5	XLM-RoBERTa	Binoculars	SuperAnnotate
Accuracy	55.94%	60.39%	80.69%	61.88%
Precision	0.72413793 0.53179191	0.58461538 0.63888889	0.72661871 0.98412698	0.74509804 0.65753425
Recall	0.20588235 0.92	0.74509804 0.46	0.99019608 0.62	0.37254902 0.48
F-1	0.32061069 0.67399267	0.65517241 0.53488372	0.83817427 0.7607362	0.49673203 0.55491329
Confusion Matrix	tn 21, fp 81, fn 8, tp 92	tn 76, fp 26, fn 54, tp 46	tn 101, fp 1, fn 38, tp 62	tn 38, fp 64, fn 13, tp 87

Table 4.2.3

4.3. Analysis of data by language

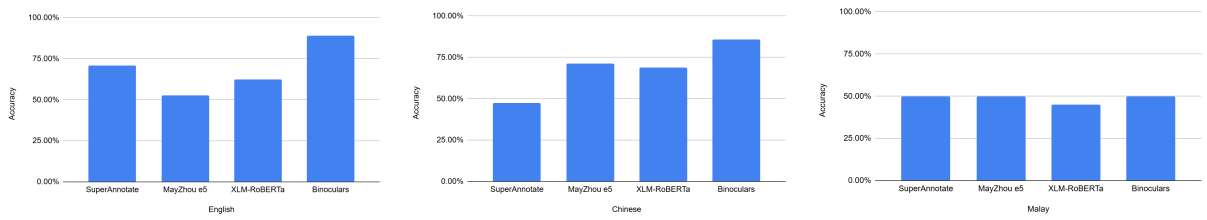
The models had varying accuracies in terms of language.

For English, Graph 4.3.1 shows that Binoculars was the most proficient at distinguishing English text with an 89.17% accuracy, while MayZhou e5 was the least with a 52.50% accuracy. For Binoculars and SuperAnnotate, English had the highest accuracy score out of English, Chinese, and Malay, suggesting that they were likely trained on more English text as compared to Chinese and Malay.

For Chinese, Graph 4.3.2 shows that Binoculars was the most proficient at distinguishing Chinese text with an 85.71% accuracy, while SuperAnnotate was the least with a 47.62% accuracy. For MayZhou e5 and XLM-RoBERTa, Chinese had the highest accuracy score out of English, Chinese, and Malay, suggesting that they were likely trained on more Chinese text compared to the other languages.

For Malay, there was the lowest accuracy score out of all three languages, with SuperAnnotate, MayZhou e5, and Binoculars scoring 50.00% each and XLM-RoBERTa scoring 45.00%. This shows that these models were trained on minimal Malay text and are unsuitable for use in detecting Malay text. Hence, it is not recommended to use these models for detecting AI text in Malay. For Malay content, it is likely that one would have to look for models specifically trained on the Malay language.

Overall, the Binoculars model had the highest accuracy across all languages and is very proficient at detecting English and Chinese text.



Graph 4.3.1 Graph 4.3.2 Graph 4.3.3

5. Binoculars

5.1. Introduction to Binoculars

The Binoculars model was trained using the zero-shot learning approach, meaning no training examples were used from the LLM source. Because of the zero-shot nature of this detector, Binoculars can spot multiple different LLMs with high accuracy, unlike many other detectors. This makes it comparable to or even better than closed-source detectors. Over a wide range of document types, Binoculars detects over 90% of generated samples from ChatGPT (and other LLMs) at a false positive rate of 0.01%, despite not being trained on any ChatGPT data. [15][16]

According to the developers, 'Binoculars' is so named as it looks at inputs through the lenses of two different language models. The researchers propose using a ratio of perplexity measurement and cross-perplexity, a notion of how surprising the next token predictions of one model are to another model. This two-model mechanism is the basis for the general and accurate detector, and how this mechanism can detect several LLMs, even when they are unrelated to the two models used by Binoculars. [16]

5.2. Accessibility

The code for Binoculars can be found on GitHub. 94.7% of the code was written in Python, with the remaining 5.3% in Shell. The code was developed and tested with Python 3.9. The installation code and usage code can be found on the Binoculars GitHub page. A limitation that Binoculars have is that it is more proficient in detecting English language text, as can be seen by the above tests conducted in section 4.3. A demo of the code can also be found on HuggingFace. According to the disclaimer, it is recommended to use 200-300 words of text at a time on the detector. Fewer words will make detection difficult, as can using more than 1000 words. Binoculars will be more likely to default to the "human written" category if too few tokens are provided. [18]

The research paper written by the developers can be found at https://arxiv.org/pdf/2401.12070. Many other researchers have also researched the Binoculars model and numerous studies on AI text detectors have included the Binoculars model in them, proving its reliability. Another commonly researched detector is DetectGPT, which was not looked at in this paper but should be considered for future works.

6. Conclusion

6.1. The current state of open-source AI text detectors

From Section 4.1, it can be inferred that the majority of open-source offline models are inadequate (or have a below 70% accuracy) at detecting GPT 4.0 text. This is likely because many of the models were trained on older versions of GPT, and many of the detectors that were

trained on GPT 4.0 text are closed source. The models that were selected were already deemed to be the top-performing models based on reviews and downloads. However, only a few of them performed up to standard. Hence, it is likely that the majority of available open-source detectors online are not suitable for use. Additionally, the top four detectors that were available were not proficient in Malay, which makes them unsuitable for use in Malay text. For English and Chinese text, Binoculars has a high accuracy rate, thus the code used for Binoculars can be examined if one were to train an AI text detector in the Singapore context.

6.2. Limitations and possible sources of error in the project

This project touches mainly on open-source detectors, however, a comparison was not made with close-source detectors due to time constraints. Additionally, there are possibly many other good open-source detector tools that this project did not manage to source for.

The dataset was a relatively small amount of only 202 and a larger dataset should be used for testing to obtain a more accurate result. The topics picked were of random selection, not following a specific category. While this ensures that there is a large variety of topics, it may not be tailored for specific types of text such as scientific reports or fake information on social media.

Only ChatGPT was used to generate text made by AI, which could lead to certain restrictions when generating texts as different chatbots have different limitations.

6.3. Future improvements

With more resources and money, the dataset should be run through closed-source detectors as well to see how well the open-source detectors perform against them. This allows for a more accurate comparison of closed-source detectors and open-source detectors. If the open-source detectors perform just as well as the closed-source detectors, the open-source detectors may have more available resources to look at for reference when developing a new AI text detector.

6.4. Key points to note when building an AI text detector based in Singapore

The three main languages to focus on training are English, Chinese, and Malay. Since the primary language of communication in Singapore is English, the bulk of the dataset should still be English text. For the most effective testing, a wide variety of topics should be covered. The model should also be trained on the newest version of GPT and be constantly updated as AI chatbots are getting updated very frequently with newer models of GPT being developed every day.

References

1. Hasan, A. (2024) *Using hugging face models in google colab: A beginner's guide*, *DEV Community*. Available at: https://dev.to/ajmal_hasan/using-hugging-face-models-in-google-colab-a-beginners-35ll

2. *Hugging face – the AI community building the future.* (no date) *Hugging Face –*. Available at: https://huggingface.co/

3. *GitHub · build and ship software on a single, Collaborative Platform* (no date) *GitHub*. Available at: https://github.com/

4. Antoun, W., Hajj, H. and Baly, F. (2020) *AraGPT2: Pre-Trained Transformer for Arabic Language Generation*, *aubmindlab/aragpt2-mega-detector-long · Hugging Face*. Available at: https://huggingface.co/aubmindlab/aragpt2-mega-detector-long

5. Solaiman (2019) *Openai-community/Roberta-base-openai-detector · hugging face*, *openai-community/roberta-base-openai-detector · Hugging Face*. Available at: https://huggingface.co/openai-community/roberta-base-openai-detector

6. *NINTW923/Bert-Lli-gptdetetor · hugging face* (no date) *Nintw923/bert-lli-gptdetetor · Hugging Face*. Available at: https://huggingface.co/Nintw923/bert-lli-gptdetetor

7. Tian, Y., Chen, H. and Wang, X. (2023) *Multiscale Positive-Unlabeled Detection of AI-Generated Texts*, *yuchuantian/AIGC_detector_env1 · Hugging Face*. Available at: https://huggingface.co/yuchuantian/AIGC_detector_env1

8. Dugan, L., Hwang, A. and Trhl'ik, F. (2024) *A Shared Benchmark for Robust Evaluation of Machine-Generated Text Detectors*, *MayZhou/e5-small-lora-ai-generated-detector · Hugging Face*. Available at: https://huggingface.co/MayZhou/e5-small-lora-ai-generated-detector

9. *NINTW923/CHATGPT-detector-LLI-HC3 · hugging face* (no date) *Nintw923/chatgpt-detector-lli-hc3 · Hugging Face*. Available at: https://huggingface.co/Nintw923/chatgpt-detector-lli-hc3

10. *Superannotate (SuperAnnotate Ai Inc..)* (no date) *SuperAnnotate (SuperAnnotate AI Inc.)*. Available at: https://huggingface.co/SuperAnnotate

11. Romero, M. (2023) *xlm-roberta-base-finetuned-HC3-mix (Revision b18de48)* , *mrm8488/xlm-roberta-base-finetuned-HC3-mix · Hugging Face*. Available at: https://huggingface.co/mrm8488/xlm-roberta-base-finetuned-HC3-mix

12. Hans, A. *et al.* (2024) *Spotting LLMs With Binoculars: Zero-Shot Detection of Machine-Generated Text*, *GitHub*. Available at: https://github.com/AHans30/Binoculars

13. Gerami, S. (2024) *Ai vs human text*, *Kaggle*. Available at: https://www.kaggle.com/datasets/shanegerami/ai-vs-human-text?resource=download

14. Sivesind, N.T. (2023) *Nicolaisivesind/human-vs-machine · datasets at hugging face*, *NicolaiSivesind/human-vs-machine · Datasets at Hugging Face*. Available at: https://huggingface.co/datasets/NicolaiSivesind/human-vs-machine

15. Hans, A. *et al.* (2024) *Spotting llms with binoculars: Zero-shot detection of machine-generated text*, *arXiv.org*. Available at: https://arxiv.org/abs/2401.12070

16. Hans, A. *et al.* (2024) *Spotting llms with binoculars: Zero-shot detection of machine-generated text*, *arXiv.org*. Available at: https://arxiv.org/pdf/2401.12070

17. Mitchell, E. *et al.* (2023) *DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature*. Available at: https://arxiv.org/pdf/2301.11305

18. Hans, A. *et al.* (2024) *Spotting llms with binoculars: Zero-shot detection of machine-generated text*, *huggingface.co*. Available at: https://huggingface.co/spaces/tomg-group-umd/Binoculars

19. Sarojag (2024) *Know about zero shot, one shot and few shot learning*, *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2022/12/know-about-zero-shot-one-shot-and-few-shot-learning/