ReviewAdvisor - Training and Evaluating Classification Models on Fake Reviews generated by Llama 2

Daniel Bauer
University of Regensburg
Daniel.Bauer@uni-regensburg.de

1 Introduction

Detecting opinion spam on websites containing user-created reviews has been an object of researcher's interest for over a decade. Most notably, Ott et al. (2011) showed that detecting deceptive opinion spam is well beyond the capabilities of human judges, demonstrating that automated approaches perform noticeably better. While they resorted to classifiers based on Naïve Bayes and Support Vector Machines (Ott et al., 2011), ever since their inception by Vaswani et al. (2017), transformer models like BERT (Devlin et al., 2018) are considered state-of-the-art in a wide variety of NLP tasks, including text classification (Sun et al., 2019). Considering the fact that tools like ChatGPT[1] have significantly reduced the barrier to produce fraudulent information, devising tools that help detect such cases has only become more and more important (Hamed and Wu, 2023).

Salminen et al. (2022), in a study that sought to train a transformer model to detect fake reviews based on a dataset containing Amazon[2] reviews, pointed out that since machine learning models face the general caveat of dataset specificity and because the nature of communication differs by platform, the applicability of fake detection classifiers across platforms should be examined. Therefore, by investigating reviews from Tripadvisor[3] instead of Amazon, I train and evaluate different classification models intended for detecting fake reviews posted on internet platforms around the domain of travel. By building on the work Salminen et al. (2022) in the form of using more modern transformer models both for generation and classification, as well as using a high-quality dataset centered around the hotel industry, I seek to provide new insights into the state-of-the-art of both machine-based review production and detection. Hence, this paper has the following goals: (1) Fine-tune a state-of-the-art generator model to produce fake reviews. (2) Fine-tune a state-of-the-art text classification model to distinguish between the reviews produced by the generator model and authentic ones. (3) Compare the state-of-the-art classifier to previous approaches. (4) Evaluate the performance of the fine-tuned classifier on human-generated reviews. Given that my project is based on the work of Salminen et al. (2022), my approach roughly mirrors theirs. Essentially, my concrete technical implementation is split into three major parts: (1) Fine-tuning a generator model to (2) produce fake reviews and (3) training a model for the detection of these fake reviews.

Additionally, I am also interested in the performance of my model when it comes to classifying human-generated "deceptive opinion spam", as described by Ott et al. (2011). Luckily, they provide their dataset, which, in addition to real reviews, contains deceptive reviews gathered from Mechanical Turk[4] and is accessible online. All of these evaluation scores will then be compared to the performance of other classification algorithms, both classical and state-of-the-art ones. The entire code of this project is supplied in this[5] Github repository.

2 Related Work

As Ott et al. (2011) already pointed out over a decade ago, reviewing products and services online has become an ubiquitous aspect of modern consumerism. Even then, websites containing consumer reviews were frequent targets of "opinion spam", as referred to by Ott et al. (2011). While research up to that point made use of primarily manual means for the identification of opinion spam, Ott et al. integrated work of psychology and computational linguistics to develop a classifier with nearly 90 percent accuracy on their opinion spam dataset. On a basic level, Ott et al. (2011) differen-

[1] https://chat.openai.com/

[2] https://amazon.com

[3] https://tripadvisor.com

[4] https://www.mturk.com/

[5] https://github.com/danielbauer1860/reviewadvisor

tiate between "disruptive opinion spam" - uncontroversial spam that is easily identifiable by a human reader, e.g. advertisements, questions, and other irrelevant text - and "deceptive opinion spam" - fictitious opinions that are intended to sound authentic. Naturally, Ott et al. were concerned primarily with the latter type. From a modern point of view, both of these types can still be found across different online platforms. However, given that the barrier to produce deceptive opinion spam has sunk significantly with the arrival of widely and easily accessible large language models, this type has arguably become only more relevant for research.

Building on the work of Ott et al. by also using their dataset, Aghakhani et al. (2018) proposed FakeGAN, leveraging the architecture of Generative Adversial Networks (GANs) for text classification. This approach consists in two models competing against each other: a generative model that tries to capture the data distribution, and a discriminative model that distinguishes between samples coming from the data or the generator model. Both of these models are trained simultaneously, the generator trying to fool the classifier, while the classifier tries to maximize its probability estimation (Aghakhani et al., 2018). Rather atypically, Aghakhani et al. (2018) do not utilize the GAN structure to create a strong generator model. Instead, they are more focused on creating a strong discriminator, with the goal naturally being to successfully distinguish between truthful and deceptive reviews as contained in the dataset created by (Ott et al., 2011). Using this approach, they managed to achieve an accuracy of 89.1 percent, on par with state-of-the-art-models of the time.

Most relevant for this paper, Salminen et al. (2022) used transformer models, both for generating the synthetic reviews, as well as for distinguishing between real and fake ones later on. While their research was primarily centered on product reviews, they showed that transformer models can quite accurately detect reviews produced by other transformer models and, thus, "machines can fight machines" in the battle against fake reviews (Salminen et al., 2022). However, their research resorted to rather outdated models, both for generation and classification. Additionally, they point out that future research is needed for experimenting with more datasets and platforms. This project intends to build and improve on both of these aspects.

3 Methods

3.1 Models for Generation

The most important decision in terms of concrete implementation is the choice of language models. For this project, this applies especially for the selection of the generator model.

3.1.1 Llama 2

Llama 2[6] (Touvron et al., 2023), which has the major advantage of being an open-source model which can be used free of charge, comes close to GPT-3.5 on academic bechmarks (Touvron et al., 2023) and is hence ideally suited for this project. It must be stated, however, that this only applies to the variant with 70 billion parameters. Given that the variant with 70 billion parameters would, once again, go beyond this project, the 7 billion variant still remains quite competitive and still clearly outperforms similarly sized variants of other open-source alternatives such as Falcon (Penedo et al., 2023) or MPT (MosaicML NLP Team, 2023) on major benchmarks (Touvron et al., 2023).

3.2 Models for Classification

As stated by Salminen et al. (2022), a model that does not share the same architecture or the same tokenizer as the generator model should ideally be chosen as a classifier. For this purpose, they resorted to RoBERTa (Liu et al., 2019). Since then, however, RoBERTa was improved upon in the form of DeBERTa (He et al., 2021), with its most recent version being DeBERTaV3 (**?**). However, I also compare its performance to other models. First, I use the RoBERTa OpenAI detector (Solaiman et al., 2019), a fine-tuned RoBERTa model designed for the detection of output generated by GPT-2 (Radford et al., 2019). Second, I also fine-tune DistilBERT (Sanh et al., 2020) in accordance to my fine-tuning of RoBERTa. Third, I also use NBSVM, as described by Wang and Manning (2012) and also used by Salminen et al. (2022). Since DeBERTaV3 builds on RoBERTa, I will first describe the properties of RoBERTa and the RoBERTa OpenAI detector. Having discussed DeBERTaV3, I elaborate on DistilBERT and NBSVM.

3.2.1 RoBERTa OpenAI detector

The base version of RoBERTa[7] (Liu et al., 2019) is based on BERT (Devlin et al., 2018) and offers a

[6] https://huggingface.co/meta-llama
[7] https://huggingface.co/roberta-base

replication of the original BERT pretraining, carefully measuring the impact of different key hyperparameters and the training data size. In doing so, they demonstrated that BERT was significantly undertrained (Liu et al., 2019). Refered to as Robustly optimized BERT approach, RoBERTa, as opposed to BERT, was trained longer, with bigger batches over more data; removed the next sentence prediction objective; was trained on longer sequences; and had the masking pattern applied to the training data changed dynamically. In combination, these modifications allowed RoBERTa to achieve state-of-the-art results in major benchmarks such as GLUE (Wang et al., 2018) and SQuAD (Rajpurkar et al., 2016). Due to these results, RoBERTa was also chosen by Solaiman et al. (2019) to be fine-tuned for detecting output generated by GPT-2, resulting in the RoBERTa OpenAI detector[8]. Recognizing that GPT-2, utilizing improved sampling methods, such as nucleus sampling, could produce seemingly credible text, RoBERTa, as a model that does not share the same architecture as GPT-2, was trained as an automated means of dealing with this issue (Solaiman et al., 2019). In precise numbers, this classifier model is able to detect 1.5 billlion parameter GPT-2 generated text with approximately 95 percent accuracy (Solaiman et al., 2019). For this project, the RoBERTa OpenAI detector was chosen to investigate how well this open-source detector model performs on detecting output generated by state-of-the-art generator models, specifically Llama 2. Since it is based on the now slightly outdated RoBERTa architecture, it also serves as a useful benchmark for the performance of the fine-tuned version of DeBERTaV3. Note that I did not fine-tune the RoBERTa OpenAI detector, since I was primarily interested in its performance "out-of-the-box".

3.2.2 DeBERTaV3

Compared to BERT and RoBERTa, DeBERTa[9] (He et al., 2021) offers two novel techniques. First, DeBERTa comes with disentangled attention. While BERT only has one vector representing one word in the input layer, which is the sum of its word embedding and its position embedding, DeBERTa divides - disentangles - this into two seperate vectors, one for the word's content and one for its position. This, in turn, entails that the attention weights are are also calculated based on two separate matrices. This division reflects the main idea of disentangled attention as stated by He et al. (2021), pointing out that the attention weight of a word pair does not only depend on their contents, but also their relative positions. The second novelty consists in DeBERTa's enhanced mask decoder. Just like BERT, DeBERTa uses masked language modeling (MLM) for pre-training. The enhanced mask decoder now involves the incorporation of absolute word position embeddings right before the softmax layer where the model decodes the masked words (He et al., 2021). These two aspects also lend DeBERTa its name: **D**ecoding-**e**nhanced **BERT** with disentangled **a**ttention. With these improvements, DeBERTa manages to consistently outperform RoBERTa across a number of benchmarks.

DeBERTaV3[10] (He et al., 2023) further improves on the architecture of DeBERTa in the form of two novelties. The first is replaced token detection (RTD) as proposed by ELECTRA (Clark et al., 2020). This method involves the use of two transformer encoders, as opposed to BERT's one. This mimics the GAN architecture, with one transformer being the generator trained with MLM, the other being the discriminator trained with a token-level binary classifier (He et al., 2023). In this technique, the generator is used to generate ambiguous tokens to replace masked tokens in the input sequence, while the discriminator needs to determine if a corresponding token is an original token or a token replaced by the generator. The second novelty comes in the form of Gradient-Disentangled Embedding Sharing (GDES), which, putting it short, provides a more efficient way of updating embeddings when compared to previous techniques (He et al., 2023). As can be seen in Table 1, DeBERTaV3, using these improvements, outperforms DeBERTa and RoBERTa in the SQuaD 2.0 (Rajpurkar et al., 2018) and MNLI-matched and mismatched (Williams et al., 2018) benchmark. For this reason, choosing DeBERTaV3 over DeBERTa and RoBERTa is the logical conclusion. This also improves on the project of Salminen et al. (2022) by implementing a more refined classification model. Further, the usage of DeBERTaV3, alongside the afore-mentioned RoBERTa OpenAI Classifier, as well as DistilBERT, described in the next section, allows for a convenient comparison between the performance of BERT/RoBERTa-based models in

[8] https://huggingface.co/roberta-base-openai-detector

[9] https://huggingface.co/microsoft/deberta-base

[10] https://huggingface.co/microsoft/deberta-v3-base

real-world applications.

3.2.3 DistilBERT

While RoBERTa, DeBERTa, and DeBERTaV3 each were intended to improve on the original BERT and their respective predecessors' architectures, DistilBERT[11] (Sanh et al., 2020) aims to provide a smaller, faster, and lighter version of BERT, while still achieving performance close to the original's. In concrete figures, this means that DistilBERT reduces the size of a BERT model by 40 percent, while retaining 97 percent of its language understanding capabilities and being 60 percent faster (Sanh et al., 2020). This is achieved using knowledge distillation (Hinton et al., 2015), a compression technique in which a compact model ist trained to reproduce the behaviour of a larger model (Sanh et al., 2020). Leveraging this technique, DistilBERT is able to achieve comparable results in both the GLUE benchmark (Wang et al., 2018) and in downstream tasks, more specifically the SQuAD benchmark (Rajpurkar et al., 2016). Further, DistilBERT comes with only 66 million parameters (compared to the 110 million of BERT-base), while having a constantly faster inference time (Sanh et al., 2020). Considering these figures, DistilBERT is a compelling option for edge applications, as well as environments with limited computational resources. For this project, DistilBERT serves as a useful comparison between state-of-the-art, large-scale BERT-based architectures (in the form of DeBERTaV3) and smaller, more light-weight options.

3.2.4 NBSVM

As described by Wang and Manning (2012), variants of Naive Bayes (NB) and Support Vector Machines (SVMs) are often used as baseline methods for text classification. SVMs are a classic algorithm that can be used across a number of different NLP tasks, due to its robust performance (François and Miltsakaki, 2012). The precise variant of SVM that was also utilized by Salminen et al. (2022), is NBSVM, as proposed by Wang and Manning (2012). They show that a simple SVM variant leveraging NB features is able to achieve consistent performance well across different tasks and datasets. Given that the other classification models are all transformer- and, more precisely, BERT-based, NBSVM was, therefore, chosen a classical baseline to provide further perspective on the performance of the other (more modern) classifiers.

3.3 Datasets

3.3.1 HotelRec

For the fine-tuning of the generator model, HotelRec[12] (Antognini and Faltings, 2020), which, at 50 million reviews, is the largest publicly available dataset in the hotel domain (Antognini and Faltings, 2020), was used. While this size may be one of the biggest advantages of this dataset, it also comes with some major practical caveats, most importantly the actual file size of the dataset being almost 50 GB in the unzipped and still 14 GB in the zipped version. Considering the scope of this project, handling a dataset of this size was simply not feasible. To add to this, the training configuration with Llama 2 only allowed roughly 3500 samples to be processed in an hour. Given the limited accessability of the free version of Google Colab, the size of this dataset, thus, had to be heavily reduced. As HotelRec is by far the largest available dataset in this domain (Antognini and Faltings, 2020) and, thereby, provides a huge variety of different reviews, reducing the data to something more manageable was still more than enough for the scope of this paper. Hence, I decided to reduce the dataset to only contain 25 000 reviews, still amounting to roughly 7 hours of fine-tuning time. Since entries were ordered by hotel, I included every 2000th entry in order to ensure a well-balanced sample data set to work with. Table 2, Figure 1, as well as Figure 2 display the most important properties of the generator training dataset created with this method.

As evident in both Table 2 and Figure 1, the reviews in the training data set that I arrived at were skewed both in terms of rating and year written. Another factor is that of review length distribution, as can be seen in Figure 2. Note that this figure only includes review lengths up to 1000 words since I only produce synthetic reviews up to this point. Additionally, in the selected sample, only 51 reviews exceed this length, making these reviews obvious outliers. Naturally, the real reviews presented to the classifiers will also exclusively include reviews under this count. Since the length distribution of the generated reviews should mimic that of the real reviews, the fake reviews were generated proportionally to the numbers displayed in Figure 2.

[11] https://huggingface.co/distilbert-base-uncased

[12] https://github.com/Diego999/HotelRec

Model	SQuAD 2.0 (F1/Exact Match)	MNLI-m/mm (Accuracy)
RoBERTa-base	83.7/80.5	87.6/-
DeBERTa-base	86.2/83.1	88.8/88.5
DeBERTa-v3-base	88.4/85.4	90.6/90.7

Table 1: Comparison of RoBERTa-based models (He et al., 2023)

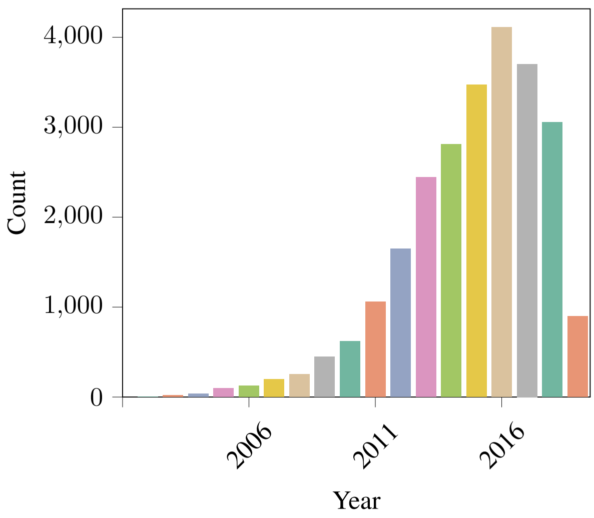

Figure 1: Distribution of Years

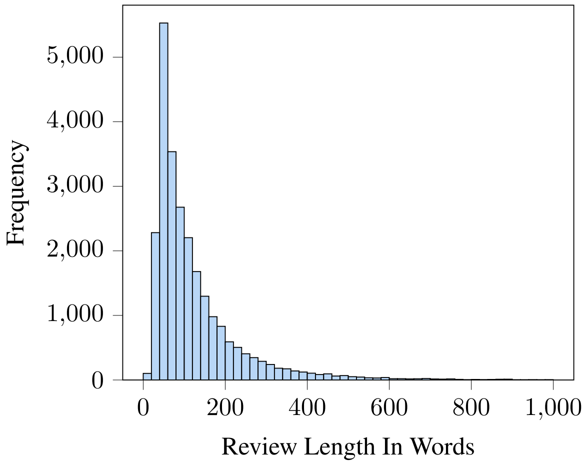

Figure 2: Distribution of Review Lengths

Rating	Count
1.0	1,219
2.0	1,233
3.0	2,829
4.0	6,730
5.0	12,989

Table 2: Distribution of Ratings

3.3.2 Deceptive Opinion Spam Corpus

The Deceptive Opinion Spam Corpus[13] (Ott et al., 2011) contains both "truthful" and "deceptive" hotel reviews of 20 Chicago hotels. While the the truthful reviews were taken from Tripadvisor (or, in the case of negative reviews, from comparable sites), the deceptive reviews were gathered via Mechanical Turk. In total, the corpus consists of 1600 reviews that can also be split according to their sentiment. Ultimately, this means that the dataset contains the following: 400 truthful positive reviews (from Tripadvisor); 400 deceptive positive reviews (from Mechanical Turk); 400 truthful negative reviews (from Tripadvisor and comparable sites); and 400 deceptive negative reviews (from Mechanical Turk). Nonetheless, for the purposes of this project, the only distinction in labels was made between truthful and deceptive. While the main focus of this paper lies in identifying fake reviews produced by large language models, this corpus allows for an interesting comparison between the classifier’s performance on synthetic and human-generated reviews.

3.4 Implementation

For the implementation of this project, Google Colab was used as a cloud service for performing the heavy calculation that comes with fine-tuning and using large language models. As already mentioned, my approach was split into three major parts: (1) Fine-tuning the generator model (Llama 2); (2) generating synthetic reviews in a stratified manner according to distribution of re-

[13] https://myleott.com/op-spam.html

view lengths using the generator model and previously extracted prompts and (3) using the generated data in combination with real data to fine-tune different classification models (DeBERTaV3, DistilBERT) or evaluate the performance of existing models (RoBERTa OpenAI detector, NBSVM). After training the model on the processed dataset, sample reviews can be generated, which will then be evaluated (Salminen et al., 2022). Subsequently, I generate the fake review dataset, which, along with the original reviews, is used as the training dataset for the classifiers. Lastly, the performance of the classification models is evaluated.

3.4.1 Training the generator

Since I fine-tuned Llama 2 via Google Colab, I first had to overcome major system constraints, namely the fact that even the 7 billion parameter version of Llama 2 does not fit into the 15 GB of VRAM that Google Colab's T4 GPU offers. As elaborated in his blog post, Norouzi (2023) solves this issue by using different quantization techniques, namely low-rank adaptation (LoRA) (Hu et al., 2021) and, more precisely, QLoRA(Dettmers et al., 2023). As shown by (Aghajanyan et al., 2020), common pre-trained language models have a very low intrinsic dimension, entailing that there exists a low dimension reparameterization that is as effective for fine-tuning as the full parameter space. Leveraging this observation, LoRA decomposes the original weight matrix into two smaller matrices, greatly reducing the number of trainable parameters. QLoRA builds on this by backpropagating gradients through a frozen, 4-bit quantized pre-trained model into low-rank adapters. Utilizing three innovations, QLoRA manages to save memory without sacrificing performance. These novelties are: a new data type in the form of 4-bit NormalFloat (NF4); double quantization, which helps reduce the average memory footprint by quantizing the quantization constants; and paged optimizers to manage memory spikes (Dettmers et al., 2023). In terms of concrete implementation, Llama-2 was quantized according to the specifications of QLoRA. More precisely, this means that Llama-2 was loaded in 4-bit, quantized using the NF4 data type while also making use of double quantization. Using this configuration, the 7 billion parameter version of Llama 2 could successfully be loaded on the T4 GPU offered by Colab. Utilizing QLoRA also required a LoRA configuration to be set up before fine-tuning. Given that the concrete implementation of the Llama 2 fine-tuning was based on Norouzi (2023), the parameters required for this configuration were also based on his approach. Both due to the scope of the paper, as well as VRAM constraints still present despite the use of QLoRA, certain trade-offs had to be made with regards to selecting the hyperparameters for fine-tuning. In concrete terms, this means the batch size was set to one, which, along with the gradient accumulation steps also being set to one, resulted in a total of 25 000 total training steps being performed. Due to the afore-mentioned constraints, only one training epoch was performed. In accordance with QLoRA, I also resorted to using a paged optimizer, which helps avoid certain memory spikes, as described by Dettmers et al. (2023). Information on the remaining hyperparameters is presented in Table 3.

Hyperparameter	Value
Batch Size	1
Gradient Accumulation Steps	1
Number of Train Epochs	1
Optimizer	Paged AdamW (32-bit)
Learning Rate	4×10^{-5}
Max Gradient Norm	0.3
Max Steps	25000
Warmup Ratio	0.03
Learning Rate Scheduler Type	Constant

Table 3: Llama 2 Fine-Tuning Parameters

Due to Colab-related limitations, the training was conducted in three sessions on a Nvidia T4 GPU with the total training time amounting to roughly seven and a half hours. The final training loss of the model was 0.39.

3.4.2 Generating the fake review dataset

Generating the reviews from the fine-tuned Llama 2 version first required appropriate prompts. In accordance to Salminen et al. (2022), these were extracted from the generator training dataset. As prompt length, I also chose five words, analogous to Salminen et al. (2022). These prompts were then randomly extracted out of 5000 reviews of the generator training dataset. The second preliminary step, calculating the length of the 5000 synthetic reviews, was also performed based on the work of Salminen et al. (2022). The length of each of the synthetic reviews is based on the length distribution of the original dataset as displayed in Figure 2. Since Llama-2 requires the length in the form of number of tokens, this figure was also calculated and later used during the generating phase. Additionally, the length of the generated reviews was allowed to up to five tokens in each direction, with the goal being to have the reviews end naturally at the end of sentences. As mentioned, the approach for generation is based on Salminen et al. (2022). Therefore, the sampling parameters used during generation were also directly taken from their implementation. In concrete terms, this means that the sampling method and the corresponding parameters mirror the work of Salminen et al. (2022). For the generation of data, a nucleus sampling approach (Holtzman et al., 2020) was selected. As demonstrated by Holtzman et al. (2020), nucleus sampling, by sampling text from the dynamic nucleus of the probability distribution, produces output that better demonstrates the quality of human text, especially compared to more traditional methods of sampling like beam search. As also pointed out by Solaiman et al. (2019), the quality of text output increases notably when employing nucleus sampling, consequently making classification of thereby produced outputs more difficult. This, however, also allows a detector trained on those outputs to transfer well across other sampling methods, proving this sampling strategy to be the most solid state-of-the-art option. The perform nucleus sampling according to the approach of Salminen et al. (2022), a top-p value of 0.92 and a temperature of 0.7 was selected. With these values Llama 2 produced 5000 reviews in a time of roughly ten hours. The final outputs then had to be post-processed, as they did not all end naturally on sentence end. Just like Salminen et al. (2022) I decided to delete final sentences that had less than three words, as they were most likely nonsensical. Sentences longer than that were kept. The processed synthetic reviews were then put together with 5000 randomly sampled reviews from the original Llama 2 training dataset to produce an evenly split training dataset for the classifiers.

3.4.3 Training the classifiers

Since the dataset is evenly split in two classes, the relevant models had to be trained for binary classification. Both DeBERTaV3 and DistilBERT were fine-tuned on the same hyperparameter configuration displayed in Table 4 as to ensure comparability between the two models.

The NBSVM training and classification was directly taken from the implementation of Salminen et al. (2022) with the only difference being that it was performed both of the sampled HotelRec and the (Ott et al., 2011) dataset. Since the evaluation goal of the RoBERTa OpenAI detector was to check the models performance on unknown data, no fine-tuning was performed on it. With the training of the classifiers complete, the next step was the evaluation of their performance.

Hyperparameter	Value
Batch Size	4
Gradient Accumulation Steps	4
Number of Train Epochs	2
Learning Rate	5×10^{-5}

Table 4: DistilBERT/DeBERTaV3 Hyperparameters

4 Results

As mentioned, the results displayed in the following must be looked at considering two caveats. First, the RoBERTa OpenAI detector was not fine-tuned at all, meaning that its evaluation metrics stem from the model's first predictions based on the respective dataset. Second, the figures on the Ott et al. (2011) dataset were similarly taken from the respective model's first shot at classification. Due to the nature of its implementation, the only exception to this is NBSVM.

4.1 Performance on Synthetic Reviews

Model	Accuracy	Precision	Recall	F1 Score
DeBERTaV3	0.9850	0.9850	0.9850	0.9850
DistilBERT	0.9765	0.9776	0.9758	0.9764
RoBERTa	0.8953	0.8845	0.9096	0.8968
NBSVM	0.9540	0.9499	0.9576	0.9537

Table 5: Generated Dataset Performance Comparison

Table 5 displays each models' performance on the generated fake review dataset with regards to the four standard evaluation metrics used for classification tasks: accuracy, precision, recall, and f1 score. For DeBERTaV3, identical values of 0.9850 are displayed for all of these metrics. This means that the model manages to make correct prediction 98.5 percent of the time, it correctly identifies 98.5 percent of the fake reviews, and it manages to find (recall) 98.5 percent of the real reviews. Naturally, the corresponding f1 score also equates to 0.985. Similarly, the DistilBERT metrics resemble each other as well. As can be noticed, all of these figures are within percentage point range of the DeBERTaV3 results. Across all metrics, it can be stated that DistilBERT reaches more than 99 percent of the performance displayed by DeBERTaV3. NBSVM, as the classical model, shows figures of 0.95 and slightly above, setting an expected and intended baseline for the transformer models. The RoBERTa OpenAI classifier, with a f1 score of 0.897, while achieving the poorest numbers out of these models still performs quite solidly, especially considering it was not fine-tuned on the dataset.

4.2 Performance on human-generated Reviews

Model	Accuracy	Precision	Recall	F1 Score
DeBERTaV3	0.5050	0.5025	0.9888	0.6664
DistilBERT	0.5038	0.5019	0.9988	0.6681
RoBERTa	0.5525	0.5295	0.9413	0.6778
NBSVM	0.8813	0.8805	0.8805	0.8805

Table 6: Ott et al. (2011) Performance Comparison

Table 6 displays each models' performance on the Ott et al. (2011) dataset containing human-generated reviews. Just like the previous section, it is also split according to the four standard metrics for this task. As becomes apparent when looking at the figures for both DeBERTaV3 and DistilBERT, both models are unable to exceed accuracy and precision metrics of 0.50 by a significant margin. When looking at the predictions made by these models, it becomes apparent that both models consistently label the Ott et al. as real, with hardly any exceptions. This also explains the high recall values. NBSVM, having been trained on the data, shows a noticeable difference in performance between synthetic and human-generated data, with seven percentage points between them. The RoBERTa OpenAI detector, at an accuracy value of 0.553 and a precision value of 0.530, outperforms both of the models fine-tuned on the generated dataset. These results will be discussed in further detail in the next section.

5 Discussion

Looking at the results of the models on the synthetic data, the performance of DeBERTaV3 immediately stands out, outperforming both other transformer models and the classical baseline in the form of NBSVM. Taking this into consideration, DeBERTaV3 appears to be a highly competitive option for this particular task, validating its choice over RoBERTa. It must be noted, however, that DistilBERT also comes close to the figures of DeBERTaV3. In fact, it manages to achieve 99 percent of its performance, despite being the more light-weight and slightly outdated option. Further, the figures of NBSVM must be considered. While being noticeably outclassed by the fine-tuned transformer models, the classical option still achieves commendable results. In combination with the metrics shown by DistilBERT, it can be concluded that all of the thus-far discussed models perform well at distinguishing between real and Llama 2-generated reviews. While DeBERTaV3 may have the best figures over-all, the other two approaches still remain as a viable option depending on the practical applications, especially if more light-weight approaches are required. The decent results of the RoBERTa OpenAI detector, in addition to being quite commendable, also suggest that output generated by Llama 2 somewhat mirrors that generated by GPT-2. Considering the performance on human-generated reviews, one must immediately point out the poor performance of DeBERTa and DistilBERT. For both of these models the figures barely manage to outperform random guesses. The fact that both models consistently declared reviews to be real also suggests that the classification models were primarily trained on the distinction between Llama 2-generated and human-generated reviews, not the discrimination between authentic and fake reviews. However, both models perform quite well in detecting human-generated reviews, as can be seen in the fact that almost all reviews were labeled as real. It must be stated, however, that Salminen et al. (2022) achieved an accuracy of 0.64 when testing their RoBERTa-based model on the dataset provided by Ott et al. (2011). This also suggests that Salminen et al. (2022) were able to produce more "human-like" reviews with their generator model than I was able to achieve. Additionally, the comparably worse results on NBSVM on the human-generated data also suggest higher feature complexity in the Ott et al. (2011) dataset when compared to the synthetic one, which was to be expected.

6 Conclusion

Three key points can be taken from the results of this project. First, DeBERTaV3 has proven itself as an outstanding model for detecting output generated by Llama 2. On a second note, however, more light-weight or classical models such as

DistilBERT and NBSVM respectively come quite close in terms of performance and remain a valid option to consider depending on practical application. Third, relying purely on data generated by Llama 2 for training the generator model has not provided satisfying results in terms of generalizability for human-generated opinion spam, at least with the applied training configuration. Especially in this regard lays the major limitation of this project, namely that of technical constraints. Ideally, Llama 2 would have been trained multiple times using different hyperparameter setups and evaluating each of them. However, given the technical limitations and the length of each training cycle, only one full run of fine-tuning was completed. It must also be noted that only the seven billion parameter version of Llama 2 was used, with the two larger models yielding potentially better results. Additionally, considering the poor generalizability of the trained classifiers, future work may fine-tune Llama 2 both on synthetic and human-generated data. Furthermore, since I based both the sampling and post-processing strategy on the work of Salminen et al. (2022), the generation parameters should be revisited and ideally tailored to Llama 2. These factors offer multiple avenues to conduct further research and should ideally be improved upon in future work.

References

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2020. Intrinsic dimensionality explains the effectiveness of language model fine-tuning.

Hojjat Aghakhani, Aravind Machiry, Shirin Nilizadeh, Christopher Kruegel, and Giovanni Vigna. 2018. Detecting deceptive reviews using generative adversarial networks.

Diego Antognini and Boi Faltings. 2020. Hotelrec: a novel very large-scale hotel recommendation dataset. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4917–4923, Marseille, France. European Language Resources Association.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Thomas François and Eleni Miltsakaki. 2012. Do NLP and machine learning improve traditional readability formulas? In *Proceedings of the First Workshop on Predicting and Improving Text Readability for target reader populations*, pages 49–57, Montréal, Canada. Association for Computational Linguistics.

Ahmed Abdeen Hamed and Xindong Wu. 2023. Improving detection of chatgpt-generated fake science using real publication text: Introducing xfakebibs a supervised-learning network algorithm.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

MosaicML NLP Team. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms. Accessed: 2023-05-05.

Armin Norouzi Norouzi. 2023. Mastering llama 2: A comprehensive guide to fine-tuning in google colab.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. *Proceedings of ACL 2011: HLT*.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Joni Salminen, Chandrashekhar Kandpal, Ahmed Mohamed Kamel, Soon-gyo Jung, and Bernard J. Jansen. 2022. Creating and detecting fake reviews of online products. *Journal of Retailing and Consumer Services*, 64:102771.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. 2019. Release strategies and the social impacts of language models.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification?

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Sida Wang and Christopher Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, Jeju Island, Korea. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.