

Exploring Data Cleaning using Language Model for Sentiment Classification Task

Puttisan "Ninja" Mukneam

May 12, 2023

Pitzer College

Overview of Data Cleaning for Machine/Deep Learning

- **Data cleaning:** The process of identifying and correcting or removing errors, inconsistencies, and inaccuracies in datasets to improve data quality and ensure reliable results.
- **Importance:** High-quality data is crucial for training effective machine learning and deep learning models, as it directly impacts their performance and generalization abilities.
- **Common issues:**
 - Missing values
 - Duplicate records
 - Inconsistent formats
 - Incorrect data types
 - Outliers and anomalies

Overview: Continue

- **Traditional cleaning techniques:**
 - Imputation
 - Deduplication
 - Standardization
 - Data transformation
 - Outlier detection and removal
- **Why do we want to clean data in Sentiment Task ?**
 - Noise Reduction
 - Standardization
 - Reducing Dimensionality
 - Handling Missing/Incomplete Data
 - Improve model performance

Literature Review: CleanML

- **Reference:** CleanML: A Study for Evaluating the Impact of Data Cleaning on ML Classification Task, by Peng Li, others (2021).
- **Robust algorithms:** Researchers have attempted to develop robust machine/deep learning algorithms that can handle dirty/noisy data without the need for data cleaning.
- **Drawbacks of robust algorithms:**
 - Limited generalization capabilities: Robust algorithms might perform well on specific types of noisy data but fail to generalize to other types or levels of noise.
 - Overfitting: These algorithms may overfit the training data, leading to poor performance on unseen data.
 - Increased complexity: Designing and implementing robust algorithms can be more complex and computationally expensive.

- **CleanML findings:** The study reveals that cleaning data before training tend to improve the performance of machine learning classifiers.
 - Higher accuracy when filling out Missing Values, cleaning outliers, cleaning mislabels, cleaning mislabels, but insignificant or even negative impact on cleaning duplicates.
 - Better generalization: Models trained on cleaned data are more likely to generalize well to new, unseen data, and can be used with many ML/DL algorithms

Justifying Transformers and Language Models for Data Cleaning

- **Context-awareness:** Transformers and language models like ChatGPT and T5 have a deep understanding of natural language and can efficiently handle context in textual data.
- **Semantic understanding:** These models can accurately identify and remove non-sentimental words from a review, preserving only the words that contribute to the overall sentiment.

Justification: Continue

- **Scalability:** Language models, when properly trained and fine-tuned, can handle data of any size and perform data cleaning tasks effectively and efficiently. This can help to scale data cleaning techniques for large and rapidly growing datasets in the Big Data era.
- **User Engagement:** With an optimized language model, the need for human intervention in data cleaning tasks can be significantly reduced. This not only increases the efficiency of the process but also mitigates the risks of human errors.
- **Semi-structured and unstructured data:** The use of machine learning tools can help detect the type of data and structure it appropriately before passing it into the cleaning model. This can help address data quality problems for semi-structured and unstructured data formats.

Justification: Continue

- **Privacy and Security Concerns:** Due to the automated nature of the data cleaning process using language models, human intervention is minimized, thereby addressing privacy and security concerns. An optimized language model doesn't require access to raw data, which can help to ensure privacy.

Language Model

Language Model

Language models are designed to understand and generate human-like text based on the input they receive. These models use deep learning techniques, particularly variants of the Transformer architecture, to process and generate text.

Once the model has processed the input, it generates a response by predicting the most probable sequence of words or tokens based on its training. It considers the context of the input, including the prompt and any previous conversation history, to produce a coherent and relevant output.

$$P_{\theta}(X_{t+1} = x_{t+1} | x_1, \dots, x_t)$$

Predicting next words given its history (previous words). And try to find θ (parameter of the model) that maximize above probability

Example

History:

I love ?

Next words:

$$P_{\theta}(? = \textit{this} | \textit{history}) = 0.4$$

$$P_{\theta}(? = \textit{game} | \textit{history}) = 0.3$$

$$P_{\theta}(? = \textit{buy} | \textit{history}) = 0.04$$

\vdots

Models

Model	Number of Parameters
gpt2	117M
gpt2-xl	1558M
t5-base	250M
t5-xl	3000M
flan-t5-base	250M
flan-t5-xl	3000M

Table 1: Pre-trained models

Transformer:

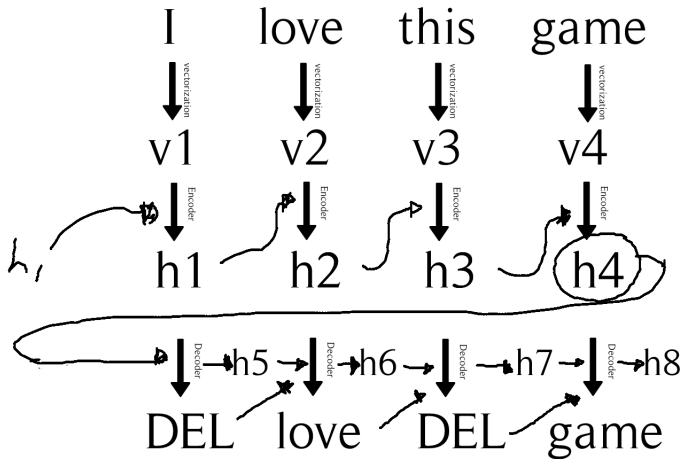
Motivation

- Limit lengths of signal paths required to learn long-range dependencies
- parallelization in training phase

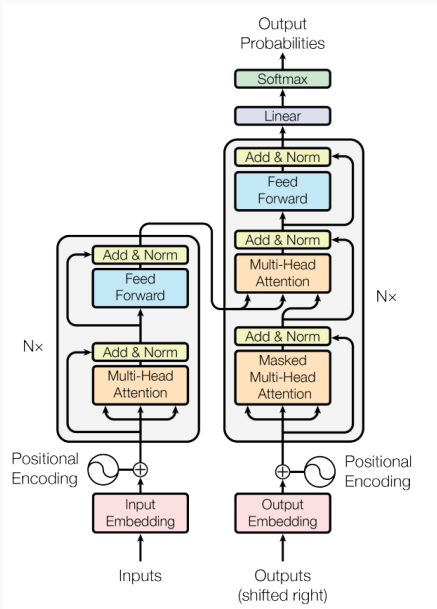
Overview

- multi-layer of "attention" generating new representation of given tokens to account for context of the whole text.

Addressing RNN



Transformer's model architecture



GPT-3: Overview

- modified transformer to be LLM
- 175 Billion parameters with 96 Layers of NN
- unsupervised trained on 500 Billion tokens ("Common Crawl", i.e., all most every (improved) text on the internet + books + wikipedia)
- Goal: given a text, generate the response
- Was supervised fine-tune for safety (Reinforcement Learning) using PPO (iterative-improvement) and reward system (from user's feedback)

- Proposed to just pre-train, then add example, then perform the task. (one-shot, few-shot) i.e., no fine-tuning

High-Level Procedure

Template := *Task Description:prompt*

Example: Without explaining, keep only words that imply sentiment of the author in an array: "I love this game"

- Tokenization/Embedding \Rightarrow [3, 200, 150, 60, ...]
- Gathering "example", i.e., previous response, etc.
- Processing Layers through self-attention outputting new representation that captures the context in the given text
- Output generation by looking at the probability distribution for the next token
- Decode \Rightarrow ['love', 'game']
- Post-processing: making sure the response is appropriate through some moderation API

T5: Introduction

Google's Text-to-Text Transfer Transformer (T5) is a highly flexible and versatile Transformer model, trained to convert text inputs into text outputs. Unlike traditional models that are fine-tuned for each specific task, T5 approaches all tasks as a text-to-text problem.

T5 views every NLP problem as a text generation task, which includes tasks like translation (text in one language to text in another), summarization (long text to short text), sentiment analysis (text to sentiment), and more.

T5: Continue

1. Both encoder, decoder
2. Open source
3. fine-tune on various NLP tasks, even better with flan-t5

Methods

Various number of reviews

Number of Reviews	Mean Squared Error	Accuracy	F-1 Score
1,000	1.08	73.0	0.791
10,000	0.824	79.4	0.843
100,000	0.629	84.285	0.883

Table 2: Midterm's model performance on testing dataset of various data size of the baseline filtered review evaluated using SVM with $C = 10$ as regularization parameters.

Creating baseline reviews

With our selected subset of 1000 reviews, we create a new column in our DataFrame named `filtered_review_text`. This column contains a filtered version of the raw review text, which is stored in the `review_text` column. We also retain the sentiment labels of the original raw reviews in a column named `review_score`.

The `filtered_review_text` column serves as our baseline for comparison with the reviews generated by the pre-trained language model. We generate this column using standard Python libraries, including `pandas`, `re`, `nltk`, and `nltk.corpus.stopwords`.

Cleaning

- Conversion of reviews into lowercase to ensure uniformity and avoid duplication based on case differences.
- Removal of all special characters using regular expressions, reducing noise in the text data.
- Elimination of stopwords (commonly used words that do not contribute to the sentiment of a sentence, e.g., 'the', 'is', 'in') to focus on words that carry sentiment.
- Stripping trailing spaces to maintain consistency in the data.
- Dropping duplicate and NAs strings value review.

Example DataFrame

	review_text	review_score	filtered_review_text
0	Nice little throw back for zombies especially ...	1	nice little throw back zombies especially sinc...
1	This game is a granddaddy its better then gta 5...	1	game granddaddy better gta tell
2	Found the game through a humble bundle sale, a...	1	found game humble bundle sale instantly fan st...
3	A truely beautiful game 11/10	1	truely beautiful game
4	Great Game. After coming to the computer versi...	1	great game coming computer version playing mob...
5	Well, I've played this game countless of times...	1	well ive played game countless times buying ga...
6	Yes, this is as bad as they say. Stay away, no...	-1	yes bad say stay away worth download
7	Awful game. 1/10 would not play again.	-1	awful game would play
8	This... this is a game... a really really bad ...	-1	game really really bad one honestly think coul...

Generating Filtered Review using Pre-trained Models

Model	Number of Parameters
gpt2	117M
gpt2-xl	1558M
t5-base	250M
t5-xl	3000M
flan-t5-base	250M
flan-t5-xl	3000M

Table 3: Pre-trained models

Prompt:

"Keeps only sentimental words: review_text"

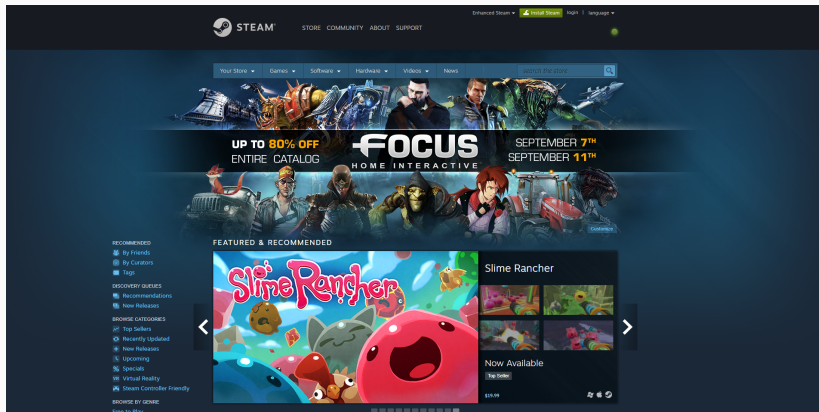
- Tokenization
 - Max_length = 50
- Generate the most likely output
 - Max_length = 25
 - early_stopping
 - repetition_penalty
- Decode

Example

	review_text	review_score	filtered_review_text	flan-t5-xl-prefix-1
0	So, do you kinda want to duel wield insane wea...	1	kinda want duel wield insane weapons kill mass...	Nazi Germany, and then you get to play the gam...
1	a fantastic game that is highly addictive, not...	1	fantastic game highly addictive bosses great f...	i'm a fan of the game and i'm a fan of the series
2	It turns boys into men, men into men, & wo...	1	turns boys men men men amp women men	It turns boys into men, men into men, & wo...
3	One of my favourite game of all time! I highly...	1	one favourite game time highly recommend playi...	a lot of new features and content.
4	good game has fun man and the girl is fun game...	1	good game fun man girl fun game many frog mann...	frog manns to apply to my fface
...
995	Boring. Only worth buying on sale if you want ...	-1	boring worth buying sale want mod something	i don't like it
996	If you are looking for fun, DON'T BUY! The car...	-1	looking fun dont buy cars handle like drunk sw...	physics are so bad that you can't even get a f...
997	Thanks for the refund. :^)	-1	thanks refund	Thanks for the refund. :)
998	Lobbies have nothing but hackers. If your look...	-1	lobbies nothing hackers looking game played ps...	I'm not sure if this is a good thing or not, b...
999	The gameplay isn't bad... if it wasn't for the...	-1	gameplay isnt bad wasnt fact micotransaction f...	a year' thing. I'm not a fan of micro transact...
1000 rows x 4 columns				

Validation

- Steam platform: largest digital distribution platform for video games



Steam Review



Not Recommended

10.3 hrs on record (7.4 hrs at review time)



POSTED: JUNE 14

I am bad at the game. But instead of taking accountability and improve my skills, I will blame the game instead.

Was this review helpful?



Yes



No



Funny



Award

239 people found this review helpful

300 people found this review funny



4



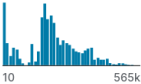


2



10

Steam Review Dataset [1]

- Approximately 6.8 million reviews.

🔗 app_id	📁 app_name	📄 review_text	# review_score	# review_votes
Game ID	Game Name	Review text	Review Sentiment: whether the game the review recommends the game or not.	Review vote: whether the review was recommended by another user or not.
	<div><div>[null]3%</div><div>PAYDAY 21%</div><div>Other (6144899)96%</div></div>	<div><div>Early Access Review16%</div><div>Early Access Review0%</div><div>Other (5392421)84%</div></div>		
10	Counter-Strike	Ruined my life.	1	0
10	Counter-Strike	This will be more of a ''my experience with this game'' type of review, because saying things like ''...	1	1
10	Counter-Strike	This game saved my virginity.	1	0
10	Counter-Strike	• Do you like original games? • Do you like games that don't lag? • Do you like games you can run on...	1	0

Text Vectorization: Bags of Words vs TFIDF

- Importance weighting: Frequency, rarity, Importance
- Reducing the impact of stopwords

TF-IDF is the short for term frequency inverse document frequency. It is a vectorizer and TFIDF-transformed data can be used directly for the classifier. The term frequency is calculated for each term in the review as

$$TF(t, d) = \frac{\text{number of times terms } t \text{ appears in document } d}{\text{total number of terms in document } d}$$

where t is the term and d is the document.

The Inverse Document Frequency is calculated by

$$IDF(t, d) = \log\left(\frac{\text{total number of documents } D}{\text{number of documents with the term in it}}\right)$$

After calculating TF and IDF with the two formulas above, the TFIDF is calculated by

$$TFIDF(t, d, D) = TF(t, d) \times IDF(t, D)$$

Multinomial Naive Bayes

Given a document d and a class label c , Bayes' theorem can be expressed as:

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)} \quad (1)$$

$$P(d|c)P(c) = P(w_1, w_2, \dots, w_n|c)P(c) \quad (2)$$

where w_1, w_2, \dots, w_n are the words in the document.

Multinomial Naive Bayes continue

$$P(w_1, w_2, \dots, w_n | c) P(c) = P(c) \prod_{i=1}^n P(w_i | c) \quad (3)$$

To classify a document, we choose the class c^* with the highest probability:

$$c^* = \arg \max_c P(c) \prod_{i=1}^n P(w_i | c) \quad (4)$$

Laplacian Smoothing

In practice, some words in the test data may not be present in the training data, leading to a zero probability for $P(w_i|c)$, which can cause issues in classification. To address this, Laplacian smoothing (also known as additive smoothing) is applied, which assigns a small non-zero probability to unseen words.

Given a word w_i and a class c , the smoothed probability is calculated as:

$$P(w_i|c) = \frac{f_{w_i,c} + \alpha}{\sum_{w \in V} (f_{w,c} + \alpha)} \quad (5)$$

where $f_{w_i,c}$ is the frequency of word w_i in class c , V is the vocabulary, and α is the smoothing parameter. A common choice for α is 1 (Laplace smoothing) or values between 0 and 1 (Lidstone smoothing).

Linear SVM

Goal: to find the optimal hyperplane that separates the data points of different classes with the maximum margin

The decision function is a linear combination of the input features:

$$f(x) = w^T x + b \quad (6)$$

where w is the weight vector, x is the input feature vector, and b is the bias term.

The goal is to minimize the following objective function:

$$\frac{1}{2}|w|^2 + C \sum_{i=1}^n \xi_i \quad (7)$$

subject to the some constrain function.

Results and Discussion

Model Comparison MNB

Model	Mean Squared Error	Accuracy	F-1 Score
Baseline	0.74	81.5	0.864
gpt2	1.0	75.0	0.821
gpt2-xl	1.1	72.5	0.808
t5-base	1.3	67.5	0.758
t5-xl	1.08	73.0	0.80
flan-t5-base	1.25	74.37	0.815
flan-t5-xl	1.14	71.356	0.78

Table 4: Models performance on testing dataset evaluated using Naive Bayes with $a = 0.1$ as smoothing parameter (fixed seed).

Model Comparison SVM

Model	Mean Squared Error	Accuracy	F-1 Score
Baseline	0.7	82.5	0.865
gpt2	1.18	70.5	0.784
gpt2-xl	1.2	70.0	0.781
t5-base	1.2	70.0	0.776
t5-xl	1.24	69.0	0.780
flan-t5-base	0.96	75.87	0.821
flan-t5-xl	1.02	74.371	0.80

Table 5: Models performance on testing dataset evaluated using SVM with $C = 1$ as regularization parameters (fixed seed).

Discussion

1. **Computational expense:**
2. **Time consumption:** Generating cleaned reviews using these models can be time-consuming.

Model	Generating time
gpt2	20m
gpt2-xl	1hr
t5-base	4m
t5-xl	25m
flan-t5-base	5m
flan-t5-xl	12m

Table 6: Approximate generating time of 1000 reviews on a free Google Colab account

Discussion: Continue

- **Accessibility:** Many state-of-the-art models such as GPT-3/4 are not open-source but are accessible via paid APIs. This can be prohibitive for small businesses or individual developers who cannot afford the associated costs.
- **Prompt design:** It can be challenging to design optimal prompts when the model has not been fine-tuned, as the training details of the original model may not be available or clear. This lack of transparency can lead to inefficiencies and inaccuracies in the data cleaning process.
- **Knowledge requirement:** Fine-tuning these models requires extensive knowledge and understanding of their architecture and parameters. Finding optimal parameters can be challenging and may require a large amount of data and computational resources. This again can be expensive and beyond the means of many users.

Future Work

- **Fine-tuning FLAN-T5 for data cleaning:** We found that fine-tuning the FLAN-T5-base model on 300K reviews with a high-end GPU (40GB memory) and 70 compute units using Google Colab Pro took more than 9 hours. However, given the flexibility of the FLAN-T5 architecture in handling various NLP tasks, we believe it holds potential for being effectively fine-tuned to perform data cleaning tasks.
- **Exploring specialized deep learning models:** We plan to investigate more optimized deep learning models that are specifically designed for data cleaning tasks. This might help in enhancing the performance and reducing the computational overhead associated with current models.
- **Fine-tuning parameters:** There is scope for exploring different fine-tuning strategies, tokenizers, and model parameters to improve the efficacy of the pre-trained models in cleaning tasks.

- **Exploring optimal prompts:** Further work can be done to investigate and design optimal prompts for the pre-trained models to maximize their performance in data cleaning tasks.

References

[1] Sobkowicz, A. (2017). Steam Reviews Dataset. Retrieved from <https://www.kaggle.com/datasets/andrewmvd/steam-reviews>

[2] Mukneam, P. (2023). sentiment_steam. Retrieved from https://github.com/pmukneam/sentiment_steam

[3] Konduru, Y. J. (2021). Game Review Classifier. Retrieved from <https://yaswanth3277.github.io/Portfolio/gamereviewclassifier.html>

[4] Kumar, S. (2018). IMDB movie review polarity using Naive Bayes Classifier. Retrieved from <https://satyam-kumar.medium.com/imdb-movie-review-polarity-using-naive-bayes-classifier-9f92c136>

[5] Reddy, V. (2018). Sentiment Analysis using SVM. Retrieved from <https://medium.com/@vasista/sentiment-analysis-using-svm-338d418e3ff1>

References: Continue

- [6] Narayanan, V., Aror, I., & Bhatia, A. (2013). Fast and accurate sentiment classification using an enhanced Naive Bayes model. Indian Institute of Technology.
- [7] Zuo, Z. (2018). Sentiment Analysis of Steam Review Datasets using Naive Bayes and Decision Tree Classifier. Retrieved from <http://hdl.handle.net/2142/100126>
- [8] Chu, X., Ilyas, I. F., Krishnan, S., & Wang, J. (2016). Data Cleaning: Overview and Emerging Challenges. SIGMOD'16, June 26-July 01, 2016, San Francisco, CA, USA. DOI: <http://dx.doi.org/10.1145/2882903.2912574>

References: Continue

- [9] Li, P., Rao, X., Blase, J., Zhang, Y., Chu, X., & Zhang, C. (2021). CleanML: A Study for Evaluating the Impact of Data Cleaning on ML Classification Tasks. 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece. DOI: 10.1109/ICDE51399.2021.00009.
- [10] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., & others. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.
- [11] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Journal of Machine Learning Research, 21(140), 1-67.

References: Continue

[12] Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., & others. (2022). Scaling Instruction-Finetuned Language Models. arXiv.

<https://arxiv.org/abs/2210.11416>

[13] Mukneam, P. (2023). sentiment-deep-cleaning. Retrieved from <https://github.com/pmukneam/sentiment-deep-cleaning>