Increase Sales from Negative Product Reviews using Bounded Response Normalized Models by Applying Safe A.I. Principles

https://www.linkedin.com/in/edy-armendariz/

Abstract. Driving an increase in sales figures is a common activity in business. Applying safe principles for Artificial Intelligence increases financial reward by automating the identification of negative product reviews and providing alternate product recommendations or advanced user workflows for the same product.

1. Introduction

Inversing the impact of negative product reviews multiplies the efficiency of building sales volumes. Mitigating known complexities in products or user workflows through detailed labels and features inserted into a machine learning training model provides three benefits. The first benefit links together negative reviews and common product support tickets with knowledge base responses. The second benefit links negative reviews with advanced user workflows, alternate product lines, or monetized workflow plugins offered by your business. The third benefit links together grammatically errant product reviews with your current foul-word database.

2. Safe A.I. Bounded Responses

Infusing your training ML data models with edge-case labels and features will create a measurable baseline for acceptable bounded responses. Re-using your foul-word database and existing data set of grammatically errant reviews increases the velocity of developing your training data models. Similarly, any mention of competitor products can be feature-mapped to equivalent data labels in your own product's offerings early in the development process. Applying a catalogue of regular expressions for removing personally identifiable information (PII) is applied at regular intervals during the ML training process and continues during live production.

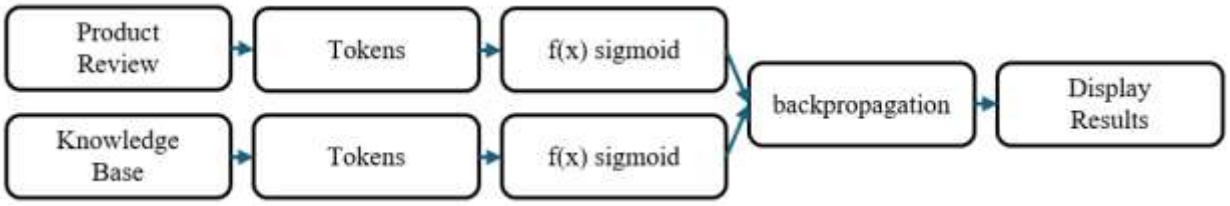
3. Model Selection

ML deployment model selection does not need to be static, locked-in, or "baked-in". The hidden layers in a neural network need not be irrevocably conjoined. Matrix columns and vector transposition (ReLU) facilitate decoupling and parameterization of outputs from one hidden layer to the input of the next hidden layer. Selection strategies can be equatable to deciding from a catalog of well documented opening strategies such as the Queen's Pawn Opening, or King's Pawn Opening from the Game of Chess. Smaller abridged ML data models can be deployed at lower cost in the cloud or in situ on the device or platform (i.e. task specific IOT and edge server

deployments). Safe AI Bounded Responses can be provided at the output of each hidden layer providing the flexibility and strategy of an Opening, Middlegame, and Endgame.

4. Linking Negative Reviews to Knowledge Base Responses

Tokenization of the natural language will include entity references to your product's name, user-interface widgets (ui-w) (including menu-options, buttons, sliders), or user-interface relative locations (ui-l) (including positions top-left, top-right, bottom-left, bottom-right). Straightforward usage of NLP with KNN will quickly align tokens from product reviews with tokens from your knowledge base.



My customers card number needs to be autofilled to save me time!!!

Product Review	
Token	Weight
My	0.025
customers	0.55
card	0.65
number	0.65
needs	0.025
to	0.025
be	0.025
autofilled	0.55
to	0.025
save	0.55
me	0.025
time	0.55
!!!	0.025

According to PCI DSS requirements, credit card sensitive Authentication Data (SAD) can never be stored unecrypted after authorization.

Knowledge Base	
Token	Weight
According	0.025
to	0.025
PCI	0.55
DSS	0.55
requirements	0.025
credit	0.65
card	0.65
sensitive	0.65
authenticatio n	0.7
data	0.55
SAD	0.7
can	0.025
never	0.025
be	0.025
stored	0.55
unencrypted	0.55
after	0.025
authorization	0.7

Display Results

ReLU Result	
Token	Weight
card	0.65
autofilled	0.55
save	0.55

5. Linking Negative Reviews to Advanced User Workflows

Re-using the same algorithm from (**4)** *Linking Negative Reviews to Knowledge Base Responses* will accelerate the linking of customer reviews to your product's regression test matrix spreadsheet used by your Quality Assurance Engineers, also known as a Requirements Traceability Matrix (RTM). Q.A. test matrices for released software products provide a better data set compared to beta software since the beta may contain technology spikes not yet ready for production. Alternative to an IDX file that can be used in (**4**), a GraphQL data structures can model vectors, matrices and facilitate transposition functions.

The GraphQL data structures for tokens and labels.

```
type Token {
score: Float
}
```

```
type Labels {
tokenVector: [Token]
}
```

The tokenized labels for the customer review, the test case description and the display results.

My customers card number needs to be autofilled to save me time!!!

Customer Review	
Token	Weight
My	0.025
customers	0.55
card	0.65
number	0.65
needs	0.025
to	0.025
be	0.025
autofilled	0.55
to	0.025
save	0.55
me	0.025
time	0.55
!!!	0.025

Plugin is required to encrypt credit card sensitive Authentication data per PCI-DSS Compliance version 3.2

QA Test Matrix	
Token	Weight
Plugin	0.025
is	0.025
required	0.025
to	0.025
encrypt	0.65
credit	0.65
card	0.65
sensitive	0.65
Authentication	0.65
data	0.55
per	0.025
PCI-DSS	0.55
Compliance	0.55
version	0.025
3.2	0.025

Display Results

ReLU Result	
Token	Weight
card	0.65
autofilled	0.55
save	0.55

The tokenized values stored into GraphQL data structures deployable in-situ.

```
Customer Review: () => ({
  tokenVector: () =>
[0.025, 0.55, 0.65, 0.65,
0.025, 0.025, 0.025, 0.55,
 0.025, 0.55, 0.025, 0.55,
0.025]
})
```

```
QA Test Matrix: () => ({
  tokenVector: () => [
0.025, 0.025, 0.025,
0.025, 0.65, 0.65, 0.65,
0.65, 0.65, 0.55, 0.025,
0.55, 0.55, 0.025, 0.025]
})
```

```
ReLU Result: () => ({
  tokenVector: () => [
0.65, 0.55, 0.55, 0.55]
})
```

6. Linking Errant Grammar to Foul-word Database

The foul word database used by your business can be updated to include instances of poor grammar and misspelled words. An in-situ deployment of foul word analysis will reduce network traffic from malicious user attacks. Deploying a smaller foul-word data filter on the client will behave as a network traffic filter that will return a mocked 200-OK status response to the malicious user without ever wasting any network resources. A more complete GraphQL, foul-word dataset filter can be deployed on the edge server, also returning a mocked 200-OK status response without consuming the origin server's resources. Traffic shaping changes in the network Queue Discipline (qdiscs) can also be inserted onto the network interface to mitigate this attack.

7. Embedded Surveys and Sales Channels

Safe AI principles can be subclassed and extended to also include baseline product quality metrics for monetization. The benefits of survey gathering exist at many junctures during the active use of a software product. For example, survey data can be gathered before and after the completion of individual product workflows to provide upgraded product capabilities at a reduced cost to the users. Telemetry can also be attached to workflows to measure the ease of use of your software product. Alternatively, users of fully paid products can optionally respond to surveys embedded in the user interface to provide constructive feedback resulting in incentivized discounts or rewards redeemable for optional workflow plugins that enhance the user experience (UX). The point is to set baseline standards and quality control for your products that piggy-back on your strategy for applying principles of Safe A.I.

8. Calculations and Code Example

The backpropagation equation for the foul-word data set can re-use the following cost function,

$$\frac{\partial \mathcal{C}0}{\partial w(\mathrm{L})} = \left(\frac{\partial z(\mathrm{L})}{\partial w(\mathrm{L})}\right)\left(\frac{\partial a(\mathrm{L})}{\partial z(\mathrm{L})}\right)\left(\frac{\partial (\mathrm{L})}{\partial a(\mathrm{L})}\right)$$

$$= \mathrm{a}^{(\mathrm{L}-1)}\sigma'\left(z^{(\mathcal{L})}\right)2\left(\mathrm{a}^{(\mathrm{L})}-\mathrm{y}\right)$$

$$\frac{\partial \mathrm{C}}{\partial w^{(\mathrm{L})}} = \frac{1}{\mathrm{n}}\sum_{\mathrm{k}=0}^{\mathrm{n}-1}\frac{\partial \mathrm{C}_{\mathrm{k}}}{\partial w^{(\mathrm{L})}}$$

These are standard loss function calculations that can be used to model your data set of tokenized foul word id values. A subset of this larger data set of foul word tokens can be delivered in-situ to the client app. For example, assuming that "crap" is a foul word that your business wants to filter, the Bert Tokenized id [11074] can be stored in the client-side cache.

```
# Python Code Example for tokenizing foul words.
import transformers
from transformers import BertTokenizer

tokenizer = BertTokenizer.from_pretrained("bert-base-cased")
tokens = tokenizer.tokenize("crap")
ids = tokenizer.convert_tokens_to_ids(tokens)
print(ids)  #prints  [11074]
```

The above Python code demonstrates the tokenization of foul words that can be added to a data set that is cached on the client-side. The tokenized values can also be cached on the edge server in JSON format.

9. Edge Server Functions

Of course, a larger data set can be stored in-situ on the edge server. At both points, malicious users can be defeated by triggering the insertion of a more aggressive traffic shaping qdisc (queuing discipline) into the network interface to protect the availability of your network.

```
/* JavaScript Code Example for blocking malicious foul-word
attacks from client requests. */
export default async (request, context) => {
      // Get the client body payload.
      const foul_words = request.body;

      // Validate using larger foul word tokenized data.
      const result = matchFoulWords(foul_words);

      if(hasFoulWords(result)){
            // Return a redirect to honeypot server.
            return Response.json({statusCode: 429, body:
            staticMsg()} )
      }
      else {
            // Return normal response.
            return Response.json({
            statusCode: 200,
            body: originServerResponse(),
            })
      }
}
```

The above code example checks the client's request body for foul words using a tokenized data set and responds back with HTTP Status Code 429, thwarting it like a DoS attack. When the client's request body does not contain foul words, we respond with happy-path dynamic content from the origin/cache server.

10. Conclusion

Applying the baseline principles of Safe A.I. also allowed us to piggy-back additional features that increase value to your business and increase the performance of your network. I have provided a tour of using pre-trained tokenized data sets on the client-side and on your edge servers that provide techniques for immediate product recommendations and protecting your network bandwidth from malicious client attacks.

References

The United States Government. (2023, October 30). *Executive order on the safe, secure, and trustworthy development and use of artificial intelligence*. The White House. https://www.whitehouse.gov/briefing-room/presidential-actions/2023/10/30/executive-order-on-the-safe-secure-and-trustworthy-development-and-use-of-artificial-intelligence/

3Blue1Brown. (n.d.). How might LLMs store facts | Chapter 7, Deep Learning. https://youtu.be/9-Jl0dxWQs8?si=Go9iJFlwiRE3dBF7&t=766

PCI Security Standards. (n.d.). *PCI DSS 3.2 Resource Guide*. pcisecuritystandards.org. https://listings.pcisecuritystandards.org/pdfs/PCI_DSS_Resource_Guide_(003).pdf

Jboback, Urban, E., & Mehrotra, N. (n.d.). *What is key phrase extraction in Azure Ai language? - azure AI services*. Azure AI services | Microsoft Learn. https://learn.microsoft.com/en-us/azure/ai-services/language-service/key-phrase-extraction/overview

www.man7.org. (n.d.). *Linux TC, show / manipulate traffic control setting*. TC(8) - linux manual page. https://www.man7.org/linux/man-pages/man8/tc.8.html#:~:text=Tc%20is%20used%20to%20configure%20Traffic%20Control%20in,for%20better%20network%20behaviour.%20Shaping%20occurs%20on%20egress

Pamula, R. S., & Gordon, C. E. (1997). *Introduction to Computer Organization and Assembly Language Programming* (2nd ed.). Irwin/McGraw-Hill.

Armendariz, E. B. (n.d.). *EdyArmendariz - Source Code Repo*. GitHub. https://github.com/EdyArmendariz