JOURNAL OF INFORMATION SYSTEMS APPLIED RESEARCH AND ANALYTICS

Volume 18, No.1
April 2025
ISSN: 1946-1836

In this issue:

iscap
Information Systems & Computing Academic Professionals

The **Journal of Information Systems Applied Research and Analytics** (JISARA) is a double-blind peer reviewed academic journal published by ISCAP, Information Systems and Computing Academic Professionals. Publishing frequency is three issues a year. The first date of publication was December 1, 2008. The original name of the journal was Journal of Information Systems Applied Research (JISAR).

JISARA is published online (https://jisara.org) in connection with the ISCAP (Information Systems and Computing Academic Professionals) Conference, where submissions are also double-blind peer reviewed. Our sister publication, the Proceedings of the ISCAP Conference, features all papers, teaching cases and abstracts from the conference. (https://iscap.us/proceedings)

The journal acceptance review process involves a minimum of three double-blind peer reviews, where both the reviewer is not aware of the identities of the authors and the authors are not aware of the identities of the reviewers. The initial reviews happen before the conference. At that point papers are divided into award papers (top 15%) and other submitted works. The non-award winning papers are subjected to a second round of blind peer review to establish whether they will be accepted to the journal or not. Those papers that are deemed of sufficient quality are accepted for publication in JISAR. Currently the acceptance rate for the journal is approximately 35%.

Questions should be addressed to the editor at editor@jisara.org or the publisher at publisher@jisara.org. Special thanks to members of ISCAP who perform the editorial and review processes for JISARA.

JOURNAL OF INFORMATION SYSTEMS APPLIED RESEARCH AND ANALYTICS

Editors

2025 JISARA Editorial Board

Are Students Willing to Pay More? An Exploration of Peer-to-Peer Payment App Use Among College Students

Philip Kim
pkim@walsh.edu
DeVille School of Business
Walsh University
North Canton, OH 44720

Austin Fantin
awfantin@gmail.com
DeVille School of Business
Walsh University
North Canton, OH 44720

Richard L. Metzer
Rlmst26@mail.rmu.edu
Robert Morris University
Moon Township, PA 15108

Abstract

When purchasing a good or service, there are now more optional payment methods than ever Recently, Peer-to-Peer (P2P) payment applications (apps) have become popular. Extant literature shows that credit cards and mobile payments have an effect on how people interact with purchases and are evaluated by pain of payment, convenience, and willingness to pay (WTP) but P2P apps haven't been evaluated using those criteria. This study seeks to fill in that gap. The study compares P2P apps with debit cards and uses cash as a constant. Surprisingly, study participants found debit cards more convenient than P2P apps for the purchase of more expensive items. However, participants were willing to pay more for a given relatively inexpensive item if allowed to use a P2P app instead of a debit card.

Keywords: Mobile payments, Peer-to-peer payments, payment transparency, willingness to pay, pain of payment

Recommended Citation: Kim, P., Fantin, A., Metzer, R. (2025). Are Students Willing to Pay More? An Exploration of Peer-to-Peer Payment App Use Among College Students. *Journal of Information Systems Applied Research and Analytics.* v18 n1 pp 4-16. DOI# https://doi.org/10.62273/ QYLZ2686.

Are Students Willing to Pay More? An Exploration of Peer-to-Peer Payment App Use Among College Students

Philip Kim, Austin Fantin and Richard Metzer.

1. INTRODUCTION

Background

Living in a competitive, continuously evolving market economy presents challenges to both providers and consumers of goods and services. Knowledge of the inner workings of the economic system can advantage individuals of either group that determine their success or failure. This study seeks to enhance the body of economic knowledge concerning how alternative payment methods affect the price consumers are willing to pay. Specifically, it extends previous work done in this area to include the use of P2P apps on mobile telephones (i.e., the Venmo app) as a method of payment. Table 1 in Appendix A provides a brief overview of the review of the literature for this study.

Payment Transparency

Soman (2003) introduced the concept of payment transparency and explored whether or not the payment method(s) offered affected the price consumers were willing to pay for goods and services. The results of the experiment showed a difference in consumer willingness to pay that could be attributed to payment transparency. Payment transparency is affected by the "salience of form, salience of amount, and relative timing of money outflow and purchase" (Soman, 2003, p.175). For example, cash had high payment transparency since physical items (salience of form) with the exact price printed on them (salience of amount) were handed over. Furthermore, since cash can run out, it presents a natural stop to spending (Boden et al., 2020). Credit cards had low transparency since consumers quickly swiped them and could have overlooked the amount they were paying. Although mobile payments or Peer-to-Peer (P2P) apps were not widely adopted during the Soman (2003) study and hence were not included, these payment tools have become increasingly popular in recent years (Lara-Rubio, et al. 2021). Since mobile P2P payment apps do not require swiping a card (salience of form), an even lower payment transparency is suggested when compared to credit or debit cards. This study seeks to explore whether or not payment transparency affects spending behavior.

Pain of Payment

Zellermayer defined pain of payment as the mental distress experienced when purchasing something (Zellermayer, 1996) and is a negative feeling that can be heightened or lowered by a variety of antecedents. Prelec and Loewenstein (1998) proposed that there is a connection between thinking about the costs and benefits of purchase and the pain of payment for the item or service. They found that thinking about the cost could reduce the pleasure of the purchase while thinking about the benefit could increase pleasure. Demand elasticity is the degree to which demand changed based on a change in price (Graham & Glaister, 2004). Demand for products with high demand elasticity change greatly in response to small movements in price. Discretionary items or seasonal consumer goods such as pumpkin spice, chai tea, and ice cream are examples of discretionary/seasonal items. Conversely, non-discretionary expenses such as fuel for transportation have less pronounced changes in demand in response to price changes. This study found that participants reported a lower pain of payment for less expensive items (i.e., chai tea and ice cream), even if the demand for the product or good was not as high. Shoes are semi-elastic. The study population needed shoes, but there were many alternatives to choose from. So, it appeared Nike Air Max shoes might be viewed as a discretionary good. Consumer behavior research (Ramya & Ali 2016) suggests that people should experience lowered pain of payment when they think of the benefits of wearing the shoes. Finally, gasoline is a highly inelastic good with respect to demand. Cars are important to society, and a majority of them are fueled by gasoline. Study participants did not shop around for gas alternatives. Instead, they saw it as a necessity. This led to thinking more about the cost, which should have raised pain of payment.

Credit Card Premium

Owning a credit card could be a rewarding endeavor. Credit cards have used multiple different structures for rewarding consumers for use, including points, sign-up and minimum spend bonuses, and travel miles (MacDonald & Evans, 2020). Furthermore, allowing consumers to purchase in the present and pay later led to a

payment float (Jalbert et al., 2010). According to the theory of the time value of money, this gave consumers more purchasing power in the present (Fernando et. al., 2021). In addition to all of these factors, credit cards are quick, easy to use, and have a smaller footprint than cash. This increases the propensity to spend more money with credit cards, aptly called the credit card premium. (Feinberg, 1986).

Debit Cards

The credit card effect has held true with debit cards as well, specifically, the fact that incentives cause consumers to spend more. Debit and credit cards share many of the same benefits, such as speed of payment, smaller physical footprint, low payment transparency, and eye-catching designs. Clerkin and Hanson (2021) investigated the credit card effect when applied to debit cards. Their study involved incentivized checking accounts, where consumers were rewarded in specific ways for high spending and punished for low spending. When compared to non-incentivized accounts, there was an 18.8% to 20.4% increase in debit card usage (Clerkin & Hanson, 2021). The increase in spending due to incentives suggests that the credit card effect applies to debit cards as well. Furthermore, this validated the use of debit cards instead of credit cards in the study.

Mobile Payments

Mobile payment usage has spiked in recent years, primarily due to increased smartphone usage (Liu & Dewitte, 2021). Smartphones are no longer used only for phone calls, texting, and email, as was the case when they first arrived on the market. Among the wide range of features they offer is mobile payments. Since many college students consistently carry their smartphones, paying via smartphone is convenient. This also partially eliminates the need to carry a wallet regularly further enhancing convenience. Mobile payments are quickly replacing payment cards as a preferred means of payment due to their convenience. Despite this, many people have yet to use their smartphones for mobile payments suggesting that mobile payments are not more convenient for everyone.

Since mobile payments have a similar payment transparency to credit cards, the credit card effect was thought to apply to them as well. This was examined in multiple ways, the main one being willingness to pay (WTP). This was the method used in this study. WTP measured how much a consumer was willing to spend for a particular good or service. In the study, WTP was examined before telling the consumer/participant what the price actually was. Liu and Dewitte (2021), examined the credit card effect on mobile payments. They found that there was not a significant effect on WTP between credit cards and mobile payments. Despite this, those who used mobile payments reported a lower pain of payment than those who used cash or a credit card.

Boden et al. (2020) furthered the research on mobile payments and suggested that since mobile payments are also charged to debit or credit cards, they should have had a similar pain of payment to cards. Phones also had the functionality to track banking instantaneously, yet they could distract from the pain due to their other apps. They hypothesized that mobile payments would only increase spending and convenience in areas where they were highly adopted. Boden et al. (2020) designed multiple purchasing scenarios to analyze WTP. A customer was primed with a specific purchasing method (cash, credit card, or mobile payment). After the priming, participants were asked their WTP for a variety of goods which changed in each study. The first study evaluated three different goods: coffee, ice cream, and a smartphone charger. It was found that higher adoption led to higher convenience, and therefore higher WTP (Boden et al., 2020). Due to this, WTP was only increased for those participants who adopted mobile payments. His second study replicated his first but applied it to a different geographical area. The same results were found here. Boden et al.'s (2020) third and final study examined WTP over four items and two price tiers, these items being ice cream, Americano, gas in a truck, and a dishwasher repair.

Since the advent of mobile payments, Peer-to-Peer (P2P) transaction apps have become popular. Venmo, CashApp, PayPal, and Zelle are among the most popular apps. These apps have become popular with millennials due to their easy availability on smartphones that millennials have widely adopted. It was found that 65% of millennials used at least one of these apps (Brown, 2017). Venmo in particular has increased its popularity by being a pseudo social media app. With users' permission, transactions and descriptions are posted on a public ledger, but monetary values are not. Caraway et al. (2017) found that the social media aspect of Venmo does not have much of an effect on how users use the app.

2. HYPOTHESES

H1: College students will find P2P payment apps more convenient than cash or debit cards, convenience being generally defined as the ease and accessibility of use.

H2: College students will be willing to pay more for low priced items with P2P payment apps than with cash or debit cards.

The hypotheses were based on the Boden et al (2020) study, where higher convenience led to a higher willingness to pay. The convenience factor was based on Soman's (2003) findings. Debit cards had a higher convenience than cash since a card took less time and effort to swipe than it did to count out bills. This study is seeking to examine if P2P apps would be even more convenient.

3. METHODOLOGY

The participants for this study were students ages 18 to 24 at a small private liberal arts university in the Midwest region of the United States. The study sought to examine undergraduate university students and therefore excluded graduate students, faculty, and staff at the university. Upon agreeing to participate, participants were randomly assigned to a cash, debit, or Venmo condition. A second part of the survey opened to those who said they had adopted a P2P app. Appendix B contains the survey, of which the conditional questions were based off the example questions from Boden et al. (2020) and adopted to my specific purchasing scenarios. The qualitative questions were developed to provide context and were not tested prior to survey release.

An online survey was created using Survey Monkey. A link to the survey was sent to the undergraduate student population at the university. Of the over 2,000 students enrolled for the fall 2023 semester, 230 survey responses were received. Of these, one student did not agree to the survey terms and did not participate. 54 participants said that they were not in the specified age and year range, and 4 left this question blank. This left 172 participants overall. There were 56 participants in the cash condition, 55 in the debit card condition, and 39 in the Venmo condition. Since Survey Monkey randomly assigned participants to these conditions, it must have been a random chance that more people assigned to the Venmo condition dropped out before the survey began.

Participants were assigned to one of three payment conditions (i.e., cash, debit card or mobile payment) before asking them about hypothetical purchasing scenarios. Since the survey focused on P2P apps, a decision to use cash, debit cards, and Venmo payment methods was taken. The use of debit cards instead of credit cards was decided upon because doing so eliminated the payment float of credit cards that could potentially skew the data. Since cash, debit cards, and Venmo were all paid immediately, these were the most comparable. Table 2 in Appendix A shows the effect that payment methods have on Willingness to Pay.

4. ANALYSIS

To analyze the data, two separate multiple regression analyses along with independent t-tests were performed. Unpaired t-tests were also utilized to evaluate the differences between WTP, convenience, and pain of payment between the debit card and P2P payment methods. Finally, Microsoft Excel's descriptive statistics tool was used to evaluate the descriptive statistics for cash, debit cards, and P2P.

Regression Analysis

The first regression evaluated convenience using price, method, and adoption. WTP responses were standardized and those scores were used in the regression analysis. For method, those who were assigned the debit card condition were coded as a "0", while those assigned P2P apps were coded as a "1". Cash was simply used as a constant. For adoption, those who signified that they used P2P apps were coded as "1", while those who did not were coded as "0." The adoption response has no relation to the method response. A respondent could theoretically have been assigned to the debit card condition yet still signified that he/she used a P2P app, thus being coded as a "0" for method yet "1" for adoption.

Higher Priced Items

Table 3 shows the results for the higher priced items. The regression examination of convenience for high priced items found a significance f of 0.0027. This was much below the expected set alpha of .05, which showed that the overall regression model was highly significant. Although multiple regression with three variables cannot be plotted on a graph due to visual constraints, multiple regression still attempts to establish a trendline.

Table 3: Results for higher priced items

	Coefficients	t Stat	p-value
Intercept	13.70572955	9.987959674	5.43445E-16
Method	-3.577702857	-3.019590502	0.003341565
Adoption	2.260444748	1.609994146	0.111106887
Z-WTP	0.453889787	0.738768802	0.462082505

The significance of the regression showed that a theoretical line does exist, therefore showing that changes in convenience were influenced by changes in method, adoption, or WTP. This set the foundation for the rest of the analyses. Furthermore, as mentioned above, the method was coded as "1" for Venmo and adoption was coded as "1" for users. Even though these are dummy variables, they are still greater in value than the other option, which is "0". When this regression was examined, it was important that these were given higher values than the null hypothesis, which stated that there was no effect on convenience for participants assigned to Venmo or non-users. Since the significance of the overall model was established, the rest of the values could be analyzed.

The multiple R, or correlation coefficient, resulted in a value of 0.3899. This showed that there was a low to medium strength linear relationship in the regression model. This meant that not much of the change in convenience could be explained by changes in the other variables. This presented an interesting situation, where the model itself was highly significant but the correlation coefficient was weak. To add to the correlation discussion, the adjusted R Square was .1221. This was much lower than expected since only about 12% of the changes in convenience could be explained by changes in method, adoption, or WTP. The difference between the significance and adjusted r-square can be explained by the theoretical trendline from the regression equation. Since dummy variables were used, there was not ever going to be a great fit of the trendline (Frost, 2018). A higher r-square in this case would have meant that the values were more clustered together with a smaller standard deviation.

The method had a p-value of .0033, which made it the only statistically significant X variable in the model. Through establishing this, it could be said that the method of payment had a statistically significant effect on convenience. This link was one of the main goals of this research, which makes this regression valuable. Therefore, the different payment conditions that students were assigned had an effect on their convenience responses. Interestingly, the t stat was -3.0195. This showed that participants reported higher convenience ratings for debit cards as opposed to Venmo. This was also the highest t stat on this model, showing that there was a strong negative effect of the method of payment. See Table 2.

Adoption had a P-value of 0.1111, which meant that it was insignificant in the regression analysis. This was unexpected, as it contradicted Boden et al.'s (2020) research that found adoption to be a highly relevant factor in convenience when related to mobile payments. WTP was also an insignificant variable with a P-value of 0.4620.

Lower Priced Items

Table 4: Results for lower priced items

	Coefficients	t Stat	p-value
Intercept	13.89503985	10.09214562	3.35239E-16
Method	-3.982427303	-3.394762077	0.001045848
Adoption	2.290637083	1.639175029	0.104873092
Z-WTP	0.821786844	1.105979348	0.271855918

Table 4 in shows the results for the lower priced items. The lower priced items (i.e., ice cream and latte) had similar regression values to the higher priced items. This showed that price did not seem to play much of a factor in the convenience ratings. For the lower priced item regression, the significance f was 0.0020, which showed that the overall regression model was significant. This also showed that the lower priced item regression was more significant than the higher priced item model. This meant that overall; the X variables had more of an effect on convenience for the lower priced items. The multiple r was 0.3984, which showed a nearly identical but slightly lower linear relationship in the data. The r square was also nearly identical at 0.1587. This was slightly higher than the higher priced items, showing that more of the variance in convenience was explained by the inputs. Due to these factors, the lower priced

items model would be considered a better model.

The method for low priced items had a P-value of 0.0010. Furthermore, the t stat was -3.3947, which was stronger directionally than the higher priced items. This was the same direction as it was for the higher priced items showing again that participants found debit cards to be more convenient than Venmo. Due to these factors, payment methods had more of an effect on convenience for the lower priced items. See Table 3.

Adoption had a P-value of 0.1048, and WTP had a P-value of .2718. This made neither of them significant as was the case for the higher priced items model. The t-stat for WTP for the lower priced items was 1.1059, while it was 0.7387 for the higher priced items. While still insignificant, this shows that participants put more weight on WTP when evaluating convenience for the lower priced items than the higher priced ones. This further shows that the X variables in the lower priced item model had more of an effect on convenience than for the higher priced items.

Descriptive Statistics

The descriptive statistics for the cash, debit card, and P2P conditions were analyzed. Since cash was a constant, this was the only analysis that could be performed on this condition. Utilizing cash as a benchmark for descriptive statistics, the difference between debit cards and P2P apps could now be analyzed. For each condition, the descriptive statistics for total convenience, total pain of payment, and total WTP were analyzed.

Cash

First, convenience for the cash condition was analyzed based on the five-point Likert scale convenience questions for the four scenarios. The mean for convenience was 10.4074, which resulted in a mean of 2.6018 per purchasing condition. This is above the halfway point on a five-point Likert scale, showing that participants found cash more convenient than not convenient. There was a minimum of 4, showing that at least one participant found all four purchases very inconvenient. Conversely, there was a max of 17. This showed that no participants found all purchases to be convenient.

Pain of payment had a slightly lower mean at 10.0925, which led to an average of 2.5231 for each purchasing condition. This was again above the halfway point, which meant that participants found that paying with cash was more pleasurable than painful. Since the average was almost exactly in the middle of the possible values, participants were overall neutral on the painfulness of paying with cash. For pain of payment, the highest value was 20. This highest value is from a five-point Likert scale for four different scenarios. This meant that at least one participant found debit cards fully pleasurable for all the purchasing scenarios.

This was double the lowest rating for the cash condition. The maximum rating was 20, which again meant that a participant found every purchasing scenario fully convenient. This high total rating was not achieved in the cash condition.

Pain of payment for debit cards also had differences when compared to cash, but not as extreme as was found for convenience. The mean was 11.1636, which is only approximately 1.5 higher than the cash condition. While the mean itself was not a meaningful difference, the standard deviations were. Cash had a standard deviation of 3.1891, while debit had a standard deviation of 5.0580. Although cash and debit cards had similar means, this showed that participants were more varied in their feelings about pain of payment for debit cards. This could have potentially been due to the adoption of debit cards, which was not a question that was asked. Everyone had used cash to pay at some point, but not everyone had necessarily used a debit card. This variance in use could lead to a variance in feelings associated with the cards.

5. RESULTS

H1: College students will find P2P payment apps more convenient than cash or debit cards.
The study did not support rejection of H1o.

The difference between debit cards and P2P was tested in convenience regression analyses and t-tests, while cash was only tested in the descriptive statistics. The convenience regressions for both high and low priced items showed that the method was the only significant variable in the model, which showed that it did have an effect on convenience. There was a negative effect, which meant that participants responded that convenience went up with debit cards. This was validated in the descriptive statistics, which showed that the mean total convenience rating for debit cards was higher than that for P2P apps. The one-tail t-test also

showed a significant difference between the means, with debit having the higher values.

H2: College students will be willing to pay more for low priced items with P2P payment apps than with cash or debit cards.
The study supported rejection of the null hypothesis.

Much like convenience, WTP was tested primarily through the regression model and t-test. For the lower priced items, multiple step-wise regression models had to be created to arrive at a significant model. Adoption and convenience were eliminated to arrive at this significant model. In this final model, the method was the only significant variable with a t-stat of 2.0142. Since this showed that the method had a positive influence on WTP, this regression resulted in rejection of the null hypothesis. When the descriptive statistics were performed, cash had the highest WTP, followed by debit cards and finally P2P apps. The t-test examining whether P2P had a higher mean WTP than debit cards was significant, but it showed that debit cards had higher means. The same happened when the test between P2P and cash was performed.

6. DISCUSSION

The main discrepancy that was found in the data was the difference in WTP between high and low priced items, as shown by the regression analyses. This was not expected, since both hypotheses predicted that WTP for P2P would be higher in both scenarios. Furthermore, the regression analyses for convenience were nearly identical, so it was assumed that the same would hold true for the WTP analysis. There were multiple potential causes for this difference between high and low priced items.

The first possible cause for this difference could have been in the app design itself. When viewing Venmo's website, splitting costs for items seemed to be its major differentiator from other payment methods. On the main page, two of the three major uses listed involved splitting costs for something. In the study, ice cream and chai lattes could easily be put on one bill, which would warrant someone paying someone else back. In contrast, sneakers were a highly individual item. One person could have bought them, and only one person could have worn them at a time. The same could be said for gas since only one car could be filled up at a time. Unless people were travelling together, gas bills were not usually split. Due to these factors, Venmo was more conducive to lower priced items. This could have led to a higher WTP for those items due to the added convenience of Venmo. Conversely, since Venmo would rarely be used for sneakers or gas, participants were not willing to pay as much with that payment method.

The other cause for the difference in WTP could be traced to the commitments needed to purchase each specific item. The lower priced items can be in-the-moment purchases without much premeditation needed, due to their low cost, time to consume, and pain of payment.

Mobile payments and Venmo behaved similarly, in which both participants need to have adopted the payment method to use it. The difference in the populations studied seemed to be the main cause for this difference. Boden et al. (2020) studied populations in the United States and India. In India, the study reported a 29% adoption rate of mobile payments as high. The United States was much lower, with Apple Pay being the most popular with only 10% adoption. The study found an 80% P2P app adoption rate.

Due to the lower adoption rates found in the Boden et al (2020) study, it would naturally be more difficult to find someone else who uses the same form of payment in order to complete a transaction. Since adoption is necessary for use, finding another user in a low adoption environment would have a large effect on convenience. As mentioned above, the study population had an 80% adoption rate. Due to this, a user would be able to assume with relative certainty that another person in the population would be a user as well. This makes adoption more of an assumption than a primary consideration, which means that it would not factor into how convenient P2P apps are. If another person in the population were not a user, then another method of payment would be found. This lack of adoption on another person's part would have cause a lowered convenience. Convenience still being a consideration was why adoption still had a relatively low P-value, albeit not a statistically significant one.

This contradicted what Boden et al. (2020) found in their study. They found that mobile payments were more convenient, but for only lower priced items (Boden et al., 2020). This showed that price did have an effect on convenience. They suggested security concerns as the reason, which if true, would have helped to explain why their findings could not be replicated. They used Amazon MTURK, which

consisted of primarily adult survey takers. College students and adults had differing views on online privacy due to the generational gap of the introduction of the Internet. This would have helped to explain much of the difference that was found.

This effect was even stronger for the lower priced items, and this could have been for multiple reasons. It was originally thought that Venmo would be more convenient, principally for the cost splitting reason in the discussion of the higher priced items. It was relatively common socially to combine a group outing into one bill.

Boden et al. (2020) found a positive relationship between convenience and WTP in their regressions. Although this same relationship was not found explicitly in the study models, debit cards were found to be more convenient than Venmo and had a higher WTP. Therefore, this implied that higher convenience relates to a higher WTP, which agreed as well with the extant literature. This disagreed with the hypothesis that Venmo would have higher convenience and WTP ratings.

Among all of the regressions evaluating WTP, adoption was the least significant X variable. This also contradicted Boden's et al.'s (2020) research, where it was found that adoption was a strong interaction with mobile payments. This could have been partially explained by the cultural changes that have happened in the few years since Boden et al.'s (2020) original study was performed. Overall adoption rates of P2P apps have only increased.

7. CONCLUSIONS

After evaluating all the regression models, it could be concluded that the lower priced items resulted in better regression models. In nearly every scenario, the f and P-values for the lower priced items were more significant than for the higher priced ones. Furthermore, this showed that a difference did exist between price levels in how college students were influenced by payment methods. This was most likely due to the differing levels of discernment required for purchasing at each price level, as previously discussed. More attention and care generally went into the decision to purchase sneakers or gas, since they had a greater effect on someone's financial well being than ice cream or latte did. This greater degree of thought could have lead to greater commitment to the purchase of a higher priced item. Once someone was committed to a purchase, the price and convenience of paying became less of a factor.

For future studies, a less homogeneous sample was recommended. The study population was drawn from a small, private liberal arts university in the Midwest region of the United States. Although the study had many more responses than expected, this homogeneous sample lowered the generalizability of the results. This study filled gaps in the existing literature, specifically surrounding college students' interactions with payment methods while introducing literature on P2P apps. Overall, it was found that college students have unique interactions with payment methods and the results seem to vary within the extant literature. In future research, it would have been interesting to expand the research regarding P2P apps to the general population, instead of solely college students.

8. REFERENCES

Boden, J., Maier, E., & Wilken, R. (2020). The effect of credit card versus mobile payment on convenience and consumers' willingness to pay. Journal of Retailing and Consumer Services, 52, 101910. Retrieved from https://doi.org/10.1016/j.jretconser.2019.101910

Clerkin, N., & Hanson, A. (2021). Debit card incentives and consumer behavior: Evidence using natural experiment methods. Journal of Financial Services Research, 60(2/3), 135-155. Retrieved from https://doi.org/10.1007/s10693-020-00334-4

Feinberg, R. A. (1986). Credit cards as spending facilitating stimuli: A conditioning interpretation. Journal of Consumer Research, 13(3), 348-356. Retrieved from https://doi.org/10.1086/209074

Fernando, J., Kindness, D., & Munichiello, K. (2021, September 3). Time value of money (TVM). Investopedia. Retrieved from https://www.investopedia.com/terms/t/timevalueofmoney.asp

Frost, J. (2018). How to interpret regression models that have significant variables but a low R-squared. Statistics by Jim. Retrieved from https://statisticsbyjim.com/regression/low-r-squared-regression/

Jalbert, T., Stewart, J. D., & Martin, D. (2010). The value of credit card benefits. Financial Services Review, 19(3), 227-244. Retrieved from https://www.financialservicesreview.com/abstracts.html

Lara-Rubio, J., Villarejo-Ramos, A. F., & Liébana-Cabanillas, F. (2021). Explanatory and predictive model of the adoption of P2P payment systems. Behaviour & Information Technology, 40(6), 528-541. Retrieved from https://doi.org/10.1080/0144929X.2020.1721228

Liu, Y., & Dewitte, S. (2021). A replication study of the credit card effect on spending behavior and an extension to mobile payments. Journal of Retailing and Consumer Services, 60, 102424. Retrieved from https://doi.org/10.1016/j.jretconser.2021.102424

MacDonald, N., & Evans, B. (2020). A theoretical examination of cash-back credit cards and their effect on consumer spending. Financial Services Review, 28(3), 223-242. Retrieved from https://www.financialservicesreview.com/abstracts.html

Prelec, D., & Loewenstein, G. (1998). The red and the black: Mental accounting of savings and debt. Marketing Science, 17(1), 4-28. Retrieved from https://doi.org/10.1287/mksc.17.1.4

Ramya, N. A. S. A. M., & Ali, S. M. (2016). Factors affecting consumer buying behavior. International Journal of Applied Research, 2(10), 76-80. Retrieved from https://www.allresearchjournal.com/archives/?year=2016&vol=2&issue=10

Soman, D. (2003). The effect of payment transparency on consumption: Quasi-experiments from the field. Marketing Letters, 14(3), 173-183. Retrieved from https://doi.org/10.1023/A:1027452904123

Zellermayer, O. (1996). The pain of paying. (Doctoral dissertation). Carnegie Mellon University. Retrieved from https://repository.cmu.edu/dissertations/351

APPENDIX A:

Table 1: Literature Review of Methods of Payment

Author(s)	Cash	Credit	Debit/Value	Mobile	Pain of Paying
Hirshman (1979)	✓	✓			
Falk et al. (2016)	✓	✓		✓	✓
Feinberg (1986)	✓	✓			
Zellermayer (1996)	✓	✓			✓
Prelec & Loewenstein (1998)	✓	✓			✓
Prelec & Simester (2001)	✓	✓			
Soman (2003)	✓	✓	✓		
Inman et al. (2009)	✓	✓			
Raghubir & Srivastava (2008)	✓	✓			
Moore & Taylor (2011)	✓	✓	✓		
Runnemark et al. (2015)	✓		✓		
Gafeeva et al. (2018)	✓		✓	✓	

Table 2: Effect of Payment Methods on Willingness to Pay

Author(s)	Mediators	Moderators	Dependent Variable	Location
Hirshman (1979)	Convenience		Basket size	US
Falk et. Al. (2016)			Store Price Image, WTP	EUR
Feinberg (1986)			WTP	US
Zellermayer (1996)			WTP	US
Prelec & Locwenstein (1998)		(Credit debt, credit line)	WTP	US
Prelec & Simester (2001)			WTP	US
Soman (2003)			Basket size	US
Inman et. al. (2009)			Basket size	US
Raghubir & Srivastava (2008)			WTP	US
Moore & Taylor (2011)			WTP	US
Runnemark et. al. (2015)			WTP	DK
Gafeeva et. al. (2018)			Recall error on spending	GER

APPENDIX B

Survey Questions

Cash Condition

- Imagine you are looking for a gas station to fill up your half-full RAM truck with 10 gal. You can only pay with cash. How much are you willing to pay for this?
- Imagine you are at a cafe and want to buy a chai latte. You can only pay with cash. How much are you willing to pay for this?
- Imagine you are at a park and are looking for ice cream. You can only pay with cash. How much are you willing to pay for this?
- Imagine you are looking for Nike AirMax sneakers. You can only pay with cash. How much are you willing to pay for this?
- How convenient would it be to pay with cash to fill up half of a tank in a RAM truck with gas?
- How convenient would it be to pay with cash for a chai latte?
- How convenient would it be to pay with cash for ice cream?
- How convenient would it be to pay with cash for Nike AirMax sneakers?
- How painful would it be to pay with cash to fill up half of a tank in a RAM truck with gas?
- How painful would it be to pay with cash for Americano?
- How painful would it be to pay with cash for ice cream?
- How painful would it be to pay with cash for Nike AirMax sneakers?
- Do you use peer to peer payment apps, such as Venmo, CashApp, PayPal, Zelle, etc.?

Debit Card Condition

- Imagine you are looking for a gas station to fill up your half-full RAM truck with 10 gal. You can only pay with a debit card. How much are you willing to pay for this?
- Imagine you are at a cafe and want to buy a chai latte. You can only pay with a debit card. How much are you willing to pay for this?
- Imagine you are at a park and looking for ice cream. You can only pay with a debit card. How much are you willing to pay for this?
- Imagine you are looking for Nike AirMax sneakers. You can only pay with a debit card. How much are you willing to pay for this?
- How convenient would it be to pay with a debit card to fill up half of a tank in a RAM truck with gas?
- How convenient would it be to pay with a debit card for a chai latte?
- How convenient would it be to pay with a debit card for ice cream?
- How convenient would it be to pay with a debit card for Nike AirMax sneakers?
- How painful would it be to pay with a debit card to fill up half of a tank in a RAM truck with gas?
- How painful would it be to pay with a debit card for a chai latte?
- How painful would it be to pay with a debit card for ice cream?
- How painful would it be to pay with a debit card for Nike AirMax sneakers?
- Do you use peer to peer payment apps, such as Venmo, CashApp, PayPal, Zelle, etc.?

Venmo Condition

- Imagine you are looking for a gas station to fill up your half-full RAM truck with 10 gal. You can only purchase this by paying a friend back with Venmo. How much are you willing to pay for this?
- Imagine you are at a cafe and want a chai latte. You can only purchase this by paying a friend back with Venmo. How much are you willing to pay for this?
- Imagine you are at a park and looking for ice cream. You can only purchase this by paying a friend back with Venmo. How much are you willing to pay for this?
- Imagine you are looking for Nike AirMax sneakers. You can only purchase this by paying a friend back with Venmo. How much are you willing to pay for this?
- How convenient would it be to pay a friend back with Venmo to fill up half of a tank in a RAM truck with gas?
- How convenient would it be to pay a friend back with Venmo for a chai latte?
- How convenient would it be to pay a friend back with Venmo for ice cream?
- How convenient would it be to pay a friend back with Venmo for Nike AirMax sneakers?

- How painful would it be to pay a friend back with Venmo to fill up half of a tank in a RAM truck with gas?
- How painful would it be to pay a friend back with Venmo for a chai latte?
- How painful would it be to pay a friend back with Venmo for ice cream?
- How painful would it be to pay a friend back with Venmo for Nike AirMax sneakers?

Qualitative Questions

- Do you use peer to peer payment apps, such as Venmo, CashApp, PayPal, Zelle, etc.? Which one(s) do you use and why?
- How many days of the week do you use one or more of these apps?
- Using a peer to peer payment app, would you be more willing to pay back a stranger or a friend?
- Imagine you owe a friend money. Would you be more willing to pay them back with cash or a peer to peer app?

Clinical Text Summarization using NLP Pretrained Language Models: A Case Study of MIMIC-IV-Notes

Oluwatomisin Arokodare
oa03242@georgiasouthern.edu
Georgia Southern University
Statesboro, GA, 30460, USA

Hayden Wimmer
hwimmer@georgiasouthern.edu
Georgia Southern University
Statesboro, GA, 30460, USA

Jie Du
dujie@gvsu.edu
Grand Valley State University
Allendale, MI, 49401, USA

Abstract

As the amount of data available in the health sector continues to grow in the era of information overload, it becomes increasingly crucial than ever to communicate essential information concisely. The vast amount of textual data from electronic health records can overwhelm healthcare professionals, reducing the time they can dedicate to patient care. A key challenge is creating comprehensive medical history summaries during patient admissions which integrate various documents including the history of present illness, discharge condition and medications, and discharge instructions. The need to address this challenge is urgent, as effective summarization of health records can greatly improve patient outcomes, enhance clinical decision-making, and facilitate access to knowledge. This study highlights the utilization of large language models trained to produce concise summaries through machine learning and national language processing algorithms. These models offer a promising avenue for summarizing patients' primary health concerns from daily progress notes, thereby streamlining information in hospital settings concisely and aiding diagnostic processes. In this study, we utilized pre-trained transformer models, including BART, T5, and Pegasus, to summarize patient medical histories. We evaluated the performance of those models using metrics including BLEU, ROUGE, and BERT scores on de-identified clinical notes from MIMIC-IV. Our experimental results show that BART and Pegasus models performed efficiently among the three large language models. The combination of these three models produced the most efficient summaries for clinical notes, given that the summary length generated by the model was shorter than the original medical history text for each medical case.

Keywords: Automatic text summarization, Natural Language Processing, Large language models, MIMIC-IV-clinical notes.

Recommended Citation: Arokodare, O., Wimmer, H., Du, J., (2025). Clinical Text Summarization using NLP Pretrained Language Models: A Case Study of MIMIC-IV-Notes Journal of Information Systems Applied Research and Analytics, v18 n1, pp-17-31. https://doi.org/10.62273/NAKA3054

Clinical Text Summarization using NLP Pretrained Language Models: A Case Study of MIMIC-IV-Notes

Oluwatomisin Arokodar, Hayden Wimmer, and Jie Du

1. INTRODUCTION

Text summarization has advanced since the 1950s to support efficient data processing in response to the growing need. It is becoming especially important in the healthcare industry, where lengthy and specialized medical reports can make it difficult to understand. The significant rise in health sector data presents challenges for healthcare practitioners that impact patient care decisions. When patients are admitted into the hospital, one of the documents written by clinical professionals to conclude their treatment is a medical history summary report. Various medical reports, including history of present illness, brief hospital course, discharge instruction, discharge medication, and general healthcare information, contribute to comprehensive record-keeping. These documents serve as valuable references for future doctors, which helps enhancing their understanding of patients' circumstances during treatment.

However, according to HealthIT.gov (2021) the official website of the United States Health Information Technology Department highlights prevailing challenges in hospitals regarding the digital exchange of health information. Text summarization can extract essential information from complex medical reports without compromising their essence. Text summarization offers a more accessible, concise understanding of information and facilitates better communication between medical experts who generate reports and patients.

In this study we use pre-trained transformer models to summarize patient medical health histories. Our goal is to evaluate their effectiveness in summarizing medical texts and perform a comprehensive analysis on the models to identify the model that can produce succinct, coherent, and precise medical history summaries.
This paper is structured into seven sections: Background information on text summarization is provided in Section 2. Related work is highlighted in Section 3. The methodology of the study is outlined in Section 4. The experimental analysis is presented in Section 5. Section 6 highlights the results discussion. Section 7 concludes with a summary of the findings and future work.

2. BACKGROUND

According to Bijal et al. (2017), text summarization is the process of condensing long text into shorter, comprehensible phrases while retaining essential information. It is crucial for managing the vast volume of online content and data including emails, movie reviews, news headlines, student notes, and more. Automation and artificial intelligence have become indispensable since they save time and provide important information so that readers may choose whether to continue or not. It plays a crucial role in regulating the deluge of information and assists users in determining whether to interact with material.

Text summarization techniques are divided into two categories: extractive text summarization and abstractive text summarization (Bhatia & Jaiswal, 2015). In this study we will be considering abstractive text summarization because it reformulates the text from the source text to generate new sentences that express the text's major concepts in a more streamlined and clear manner. Unlike extractive summary, which chooses phrases straight from the source text to generate a summary, abstractive text summarization will rephrase and condense subject matter resulting in summaries that are simpler to read and comprehend while retaining overall meaning (see Figure 1). Gaikwad and Mahender (2016) generated sentences using keywords with the aim of minimizing redundancy and produce accurate summaries. To accomplish this, a comprehensive grasp of the text's context and significance is imperative, along with the ability to rephrase and paraphrase it without compromising its essence. Natural language processing (NLP) interprets and understands the content of a document or text to generate an abstract text summary. Abstract summarization, though capable of generating concise and coherent summaries, typically poses greater

challenges due to the requirement for advanced natural language generation techniques.

3. LITERATURE REVIEW

This section discusses the relevant literature used as the basis of our methodology. Specifically, these studies address the use and combination of the abstractive large language models for text summarization.

In the context of the increasing amount of online content, Batra et al. (2020) discussed the importance of text summarization tools. These tools provide concise summaries, which allows readers to decide whether to dig deeper into the content or not. There is a growing need for text summarization to handle complex language as the volume of information on the Internet is increasing and it is increasingly difficult to extract relevant information manually. To assess their effectiveness, popular techniques such as the Encoder-Decoder Model with Attention, the Pointer Generator, the Pointer Generator with Coverage, UniLM, and BERTSUMABS are analyzed. As part of the study, UniLM is highlighted as one of the models examined using ROGUE metrics. These metrics are applied to the CNN/Daily Mail dataset, and the results are compared with reference summaries. For each model, ROGUE metrics were used to evaluate its results (ROGUE 1, ROGUE 2, ROGUE L), and the scores were compared on CNN/Daily Mail datasets showing overlap and common subsequence statistics.

Tsai et al. (2022) tackled privacy concerns and lack of public datasets in studying outpatient conversations. To address this, they proposed a three-step framework for summarizing outpatient conversations using Transformer-based models and external medical data. The long outpatient conversions are summarized through dialogue segmentation, dialogue summarization, and writing style conversion. The Multilingual T5 (mT5) model was used to summarize longer inputs despite limited training data. The technique yields steady performance in various tasks, as demonstrated by the experimental findings using pre-trained models.

Van Veen et al. (2023) evaluated eight Large Language Models (LLMs) for clinical text summarization from electronic health records (EHR). The experiments included quantitative evaluations and a clinical reader study to assess LLM performance and potential improvements in healthcare workflows. Adaptation methods are highly important in the study, which shows that even one in-context example significantly improves performance. When sufficient in-context examples are provided, proprietary models GPT-3.5 and GPT-4 consistently outperform open-source models. In all metrics across datasets, Sequ2seq models (FLAN-T5, FLAN-UL2) outperform autoregressive models (Llama-2, Vicuna), with GPT-4 achieving the highest performance on all metrics. FLAN-T5 excels in syntactical metrics. The results show that LLM-generated summaries often surpass human experts in completeness, correctness, and conciseness.

H. Zhang et al. (2019) introduced a neural network framework with an encoder-decoder architecture for summarizing multiple sentences in a document. The two-stage encoder-decoder framework combines BERT to encoding input sequences and Transformer-based decoding to predicting words sequentially. The model uses pre-trained contextualized language models to enhance performance without manual features, maximizing the likelihood of generating accurate summaries. The model demonstrates improved performance on CNN/Daily Mail and New York Times datasets, achieving state-of-the-art performance on CNN/Daily Mail with a score of 33.33 on ROUGE-1, ROUGE-2, and ROUGE-L, and a 5.6% relative improvement in ROUGE-1 on the New York Times dataset. The NYT50 corpus generates longer summaries than CNN/Daily Mail, and the model captures long-term dependencies effectively. The model performs better across diverse data distributions than other methods, with significant improvements observed in ROUGE-1 and 0.51 in ROUGE-2.

Yang et al. (2022) highlighted the significance of NLP powered by clinical language models and focused on utilizing artificial intelligence (AI) for processing digital health records. They presented GatorTron, a huge clinical transformer model based on a corpus of more than 90 billion words from UF Health, PubMed, the website Wikipedia, and MIMIC III. When tested on five clinical Natural Language Processing (NLP) tasks, GatorTron trained with different parameter sizes consistently outperformed previous clinical and biological transformers. The findings suggest that increasing the number of models and training data can greatly enhance medical AI system performance, which may have consequences for the provision of healthcare.

The advancements in neural network technologies and the availability of large

amounts of data are responsible for the rise of summarization models in information technology (Kryściński et al., 2019). The methods used nowadays include hybrid extractive-abstractive models, multi-task training, copying mechanisms, attention mechanisms, and reinforcement learning. Despite these developments, benchmarks such as the CNN/Daily Mail news corpus have not advanced as much as they formerly did. Uninformative assessment processes and uncurated datasets in research setups are to blame for this stagnation. A more reliable setup for text summarization, with special emphasis on analyzing datasets, assessment measures, and model outputs is needed. Large-scale summarizing model assessment is labor-intensive whether done manually or semi-automatically. This has prompted the creation of automatic measures like the ROUGE package, which evaluates the degree of lexical similarity between prospective and reference breakdowns.

4. METHODOLOGY

In our study, we conducted experiments on distinct datasets MIMIC-IV-Note to assess the performance of various abstractive Large Language Models (LLMs) in the context of text summarization. These datasets serve as the foundation for our evaluation and comparison of the summaries generated by large language model. Figure 2 appendix Item captures the framework of our methodology. We employed pre-trained transformer models, including BART, T5, and Pegasus, which were used to summarize patient medical histories from the dataset. We evaluated the performance of those models using standard metrics presented in the LLM literature which includes BLEU, ROUGE, and BERT with the equations for their respective calculations detailed in the experimental results. Our experimental results show that BART and Pegasus models performed efficiently among the three large language models. The combination of these three models produced the most efficient summaries.

Dataset

The dataset used in this research is MIMIC-IV-Note: de-identified free-text clinical notes. According to Johnson et al. (2023), the dataset contains 357,289 discharge summaries and 2,471,881 radiology reports from 161,403 patients admitted to the Beth Israel Deaconess Medical Center in Boston, MA, United States. The dataset belongs to the Medical Information Mart for Intensive Care (MIMIC), and it has protected health information removed in accordance with HIPAA Safe Harbor provisions. The dataset consists of unstructured text data, and it is intended to stimulate research in clinical natural language processing and related areas, providing context to the clinical data within the MIMIC-IV database (Johnson et al., 2018). It includes a diverse set of clinical notes, which include a wide range of medical information such as patient present illness histories, discharge conditions and instructions, diagnoses, discharge medication, and treatment plans.

Preprocessing of the MIMIC-IV-Clinical Note Text Dataset

The dataset contains information about patient discharge for hospitalizations. These are long form narratives which describe the reason for a patient's admission to the hospital, their hospital course, their health history, and any relevant discharge instructions. In this study, we focused on the medical health history of patients.

According to Johnson et al. (2018), the steps in the preprocessing involve:

1) eliminating empty and/or duplicate clinical notes, converting all text to UTF-8 encoding, and removing any invalid UTF-8 sequences,
2) standardizing special characters,
3) tokenization-dividing the medical text into smaller units like words or phrases,
4) performing normalization - we ensured text uniformity by converting it to lowercase and removing unnecessary spaces and expanding contractions,
5) performing lemmatization to reduce the words to their base form to handle variations,
6) assigning grammatical categories to words and grouping them based on their grammatical structure,
7) removing details that are irrelevant to the analysis or could potentially identify the patient, and masking any identifiers that could link the data back to a specific patient,
8) ensuring consistent tokenization so that the same entities are consistently anonymized throughout the dataset, and
9) ensuring that the de-identified data comply with relevant privacy regulations such as Health Insurance Portability and Accountability Act (HIPAA) while retaining the useful information for the analysis.

Large Language Models

Abstractive approach allows for enhanced comprehension and coherence. This is

particularly important in clinical contexts where the summaries need to be easily understandable by medical professionals. Therefore, we focus on the abstractive models in this study. We aim to evaluate and assess the efficacy of three finely tuned state-of-the-art abstractive text summarization models: BART, T5, and Pegasus, each of which was trained on our dataset.

A detailed overview of the three LLMs employed for the patient history clinical text summarization is presented in this section. These models represent advanced, large-scale NLP models that can understand and generate human-like language using complex machine learning techniques and extensive training on text data.

Bidirectional and Auto-Regressive Transformers (BART) Model

The BART model is a type of transformer-based neural network architecture introduced by Facebook AI Research and is designed mainly for text generation, summarization, and translation tasks. In 2019, the BART model combines two popular architectures elements: BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer) models, enabling it to be fine-tuned on small, supervised datasets for domain-specific tasks (Devlin et al., 2018). It generates autoregressive sequences with an autoregressive decoder and records contextual information from each side of a sequence using a bidirectional encoder. BART model is more effective than BERT and GPT-1, with 140 million parameters, because of its special mix of autoregressive generation and bidirectional context awareness. The encoder-decoder mechanism that makes up BART's architecture is used to mask or remove input tokens during preprocessing, which results in an inaccurate representation of the sequence (Arokodare & Wimmer, 2023). Subsequently, the corrupted form of the sentence is rebuilt, the corrupted input is mapped to a latent representation by the encoder, and the original phrase is generated by the decoder using this representation.

Text-to-Text Transfer Transformer (T5) Model

As introduced by Google AI researchers in 2019, the Text-to-Text Transfer Transformer (T5) model is a transformer-based neural network architecture. All tasks are framed as converting one textual input into another textual output, which is called a "text-to-text" approach. It ensures accuracy across tasks by learning to translate input and by minimizing a loss function. According to Roberts et al. (2019), transfer learning is a potent technique designed for a variety of natural language processing tasks which involves pre-training a model on a data-rich task before fine-tuning it for downstream tasks. T5 is trained on a range of tasks and datasets using a unified text-to-text architecture. By providing appropriate input-output pairs during training, it can handle a broad variety of tasks and obtain state-of-the-art results throughout a wide range of language comprehension tasks, such as translation, summarization, question answering, text classification etc. T5 has encoder and decoder layers and is comparable to those of BERT and GPT. It has high reliability and adaptability and has been utilized in many benchmarks and applications.

Pegasus Model

In 2020, Google AI researchers developed the transformer-based neural network model, PEGASUS. Its purpose is to produce precise and succinct summaries of lengthy papers or articles. It has been tailored for abstract text summarization. PEGASUS expects masked sentences from an input document using a pre-training task known as "gap-sentence generation" and a self-attention mechanism. This enables the model to comprehend sentence interactions and provide logical summaries depending on the context from which they originate. According to Delangue (2016), PEGASUS creates summaries by rewriting the original text in a way that keeps consistency while collecting the essential details. PEGASUS is a sequence-to-sequence model with a similar encoder-decoder architecture to BART. It is pre-trained using Masked Language Modeling (MLM) and Gap Sentence Generation (GSG), both of which use a causal mask to hide future words. MLM randomly replaces encoder input tokens, while GSG replaces entire sentences with a mask, like a regular auto-regressive transformer decoder.

Abstractive Combined Model

In the context of clinical text summarization, a combined model is a language model that combines several features of multiple pre-trained abstractive models to provide comprehensive summaries of clinical documents. The goal is to harness each model's distinct unique strengths and capabilities as an abstractive text summarization tool and to provide summaries that are more precise, thorough, and more accurate.

Insights into Model Selection
In this study we considered some rationales like architecture, performance, and adaptability for managing the complexity of clinical text which are crucial when choosing models for clinical text summarization. However, each model (BART, T5, and Pegasus) are effective clinical text summarization models, each with its own strengths and challenges.

- **BART model** is designed to handle a variety of NLP tasks and to combine the strengths of both bidirectional and autoregressive models. The architecture of this model requires significant computational power and memory. It is useful for summarizing a variety of noisy, unstructured text and messy clinical notes.

- **T5 model**'s text-to-text format enhances the accuracy of clinical reports, despite being extremely CPU-intensive. Its consistent methodology results in excellent quality summaries spanning different tasks.

- **Pegasus model** is a pre-trained model that demonstrates its efficacy in managing the complicated and relevant structure of clinical text through the generation of clear and pertinent summaries. Pegasus, like BART and T5, requires significant computing capacity.

We selected the **combination of the 3 large language models** to summarize the clinical text because clinical data often contains complex medical terminology and detailed patient information.

- The combined summarization model will provide a summary that captures the underlying medical information & context more effectively.
- The combined abstractive model summarization synthesizes key points, reducing redundancy and emphasizing relevant information. This ensures clinical summaries are concise and focused on critical aspects of a patient's health and treatment, which is essential for efficient clinical decision-making.
- The combination of these three models allows us to expand the overall performance and quality of our summaries provided by the models.

5. EXPERIMENTAL ANALYSIS

Hardware and Environment Setting
The experiment and testing procedures presented in this paper were conducted using a Dell Inspiron 14 mounted with the Windows 11 operating system and the processor is Intel® Core i7-1255U CPU @1.70 GHz. The MIMIC-IV dataset was imported into the Python Google collab notebook, a platform optimized for swift Python coding. Given the dataset's substantial size, we opted for GPU runtime to enhance processing efficiency. For each large language model variant, essential libraries such as AutoTokenizer and pipeline dependencies from transformers were installed from the Hugging Face community package. These packages, offered by the Hugging Face community, provide tools for building, training, and deploying open-source machine learning models and ensuring accuracy and effectiveness throughout the summarizing procedure (see Appendix Item Figure 3). The model parameters used include "facebook/bart-base," "t5-base," and "google/pegasus-large" pre-trained models for BART, T5, and Pegasus, respectively. The necessary auto tokenizer and pipelines dependencies from transformers for BART model, BartForConditionalGeneration and BartTokenizer were utilized, while T5 model employed T5ForConditionalGeneration and T5 Tokenizer. Pegasus variants employed Pegasus Tokenizer and PegasusForConditionalGeneration. The selection of pre-trained models was informed by memory constraints in Google Collab.

Evaluation Metrics/ Performance
In this section, evaluation metrics employed in the text summarization experimentation are analyzed. These metrics serve to assess the quality and effectiveness of the generated summaries, leveraging a suite of widely recognized and accepted evaluation criteria of different LLMs.

BLEU Score (Bilingual evaluation understudy)
According to Papineni et al. (2002), BLEU is a metric for assessing the quality of text translated by a machine from one natural language to another. The algorithm uses n-grams found in human-translated sentences. It measures the similarity and the precision of the model's output compared to a reference summary.

The geometric mean of modified precision scores in a test corpus is calculated, multiplied by an exponential brevity penalty factor, and then used to compute the brevity penalty BP and

weighted by the BP. The formular is shown below:

$$P_n = \frac{Count\ of\ n-grams\ in\ translation\ that\ appear\ in\ refernce}{Count\ of\ n-grams\ in\ generated\ translation} \quad (1)$$

$$BP = \begin{cases} 1 & if\ c > r \\ e^{(1-r/c)} & if\ c \leq r \end{cases} \quad (2)$$

$$\text{BLEU Score} = BP \times \exp\left(\sum_{n=1}^{N} WnlogPn\right) \quad (3)$$

Where:

- c is the length of the generated translation that appears in reference
- r is the length of effective reference corpus length.
- Wn is the weight assigned to the precision of n-grams.
- Pn is the precision of n-grams.

The BLEU score ranges from 0 to 1. Higher BLEU scores indicate better overlap between the generated summary and the reference summary, indicating better quality translation. Lower BLEU scores imply less precision or accuracy in the model's output compared to the reference summary, indicating lower quality translation.

ROUGE Score (Recall-Oriented Understudy for Gisting Evaluation)

According to Lin (2004), ROUGE Score assesses the overlap of n-grams (sequences of words) between the generated summary and reference summaries. It considers metrics such as ROUGE-N (unigrams, bigrams, etc.) and ROUGE-L (longest common subsequence) to evaluate content overlap.

ROUGE-N refers to the overlap of n-grams between the system and reference summaries.

$$ROUGE - N = \frac{Total\ number\ of\ n-grams\ in\ reference\ summary}{Number\ of\ overlapping\ n-grams} \quad (4)$$

ROUGE-1 is the term used to describe how the framework and reference summaries overlap in terms of unigrams, or words.

$$ROUGE - 1 = \frac{Total\ number\ of\ unigrams\ in\ reference\ summary}{Number\ of\ overlapping\ unigrams} \quad (5)$$

ROUGE-2 refers to the overlap of bigrams between the system and reference summaries.

$$ROUGE - 2 = \frac{Total\ number\ of\ bigrams\ in\ reference\ summary}{Number\ of\ overlapping\ bigrams} \quad (6)$$

ROUGE-L refers to statistics based on the Longest Common Subsequence. To automatically identify the longest co-occurring in sequence of n-grams, the longest common subsequence issue takes sentence-level structural similarity into consideration.

$$ROUGE - L = \frac{Total\ number\ of\ words\ in\ reference\ summary}{Length\ of\ longest\ common\ subsequence} \quad (7)$$

The ROUGE metrics indicate that scores between 0 and 1. Higher ROUGE scores indicate better recall of important content from the reference while low ROUGE indicates that the generated summary may not accurately capture or recall important content from the reference summary.

BERT Score (Bidirectional Encoder Representations from Transformers)

A BERT Score measures the similarity between the model's representation of the summary and the reference using pre-trained contextual embeddings (T. Zhang et al., 2019). The formula computes the cosine similarity between the generated and reference phrases' contextual embeddings (Hanna & Bojar, 2021):

$$\text{BERT Score} = \sum_{i=1}^{L} \mathbf{CS}(\boldsymbol{xi}, \boldsymbol{y})\ /\boldsymbol{L} \quad (8)$$

$$\text{BERT Score} = \sum_{i=1}^{N} \mathbf{F1}(\boldsymbol{CS}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{i}))/\boldsymbol{N} \quad (9)$$

Where:

- CS (xi, y) is the cosine similarity between a generated sentence x and its entire reference sentence y based on the contextual embeddings for the i-th token in each.
- L is the length of the generated sentence.
- N is the number of layers of BERT used for scoring.
- F1 is the harmonic mean function.

Higher BERT scores indicate a closer semantic match between the generated and reference summaries. Low BERT Scores suggest that the generated summary may not closely match the meaning or content of the reference summary.

6. DISCUSSION

The experiments focused on summarizing three test samples, each corresponding to a clinical note, from the MIMIC-IV-de-identified dataset. Performance evaluation includes metrics like BLEU, ROUGE, and BERT scores. The experimental results demonstrate the summarization capabilities of three large language models across diverse patient clinical notes and the evaluation metrics used in these

experiments are examined to gauge the quality and efficacy of the generated summaries. Evaluation performance of the three LLMs for text summarization is detailed in Appendix Tables 1 - 3. The experimental results reveal distinct strengths among the three LLMs evaluated. The analysis showcases these models' impressive capacity to summarize complex medical reports into more concise forms.

In the first clinical note, the table performance scores show the comparison between the individual models and the combined model. The BART model demonstrates superior precision, achieving the highest BLEU score (0.012046), similar ROUGE-1 and ROUGE-L scores (0.304348), and the highest ROUGE-2 score (0.299270) across the 3 models. This suggests strong alignment between the generated and referenced summaries in terms of unigram overlap, longest common sequence, bigram overlap, and BERT score.

In the second and third clinical notes, the table performance scores show the individual model scores, the Pegasus model outperforms the other models in evaluation metrics including BLEU score, ROUGE score, and BERT scores. This indicates superior precision, recall, and F1 score in summary generation and suggests strong alignment between the generated and referenced summaries regarding unigram overlap, longest common sequence, and bigram overlap.
However, from the tables of evaluation performance, the combination of the 3 models shows significantly higher scores across all metrics in each clinical case sample indicating better overall performance and outperforming the individual models. The combined models indicate that combining the features from the three models will lead to better performance in generating a concise summary of clinical notes.

A comparison between the original clinical note and the summaries generated by the three LLMs and the combined model is shown in Appendix Tables 4 - 6, which echoes the evaluation performance reported in Appendix Tables 1 – 3. The summary generated by the combined model seems to contain more comprehensive and essential information from the original text source compared to three individual LLMs. The experimental findings and the generated text summaries underscore the remarkable proficiency of the three large language models when combined.

7. CONCLUSIONS AND FUTURE WORK

This study evaluates the performance of three widely used abstractive large language models (BART, T5, and Pegasus) and a combination of these models for clinical text summarization. The experimental results highlight the distinct strengths of each model, with the combined model emerging as the most effective approach for summarizing clinical notes. Limitations include a limited dataset. Furthermore, there are additional LLMs which need to be included in the evaluation. Additionally, in future work we aim to introduce the gold standard of human evaluation.

The combined model consistently outperforms the individual models across BLEU, ROUGE, and BERT metrics, demonstrating its superior ability to produce high-quality and robust summaries of clinical information. As detailed in Appendix Tables 1-3, integrating BART, T5, and Pegasus leverages the unique strengths of each model, resulting in a more comprehensive and accurate summarization.

The Implications of these experimental results are significant for clinical practice includes:

- **Reduced Cognitive Load**: Healthcare professionals can spend less time interacting with patients and providing urgent treatment when the cognitive strain of reading through extensive clinical notes is lessened. In demanding settings like critical care facilities and emergency departments, this is extremely advantageous.

- **Consistency and Accuracy:** The combined model ensures precise and consistent summaries by integrating many evaluation metrics. Such consistency is particularly important in clinical settings, where discrepancies or missing information in data can have serious implications.

- **Improved Efficiency:** With the combined model, healthcare professionals can quickly grasp essential information and review extensive medical records in less time and with greater efficiency. Precise summaries of medical text enable physicians to swiftly assimilate important details, leading to improved treatment of patients.

- **Improved Decision-Making Process:** To improve diagnostic and medical care decisions, the combined model ensures that medical practitioners have access to top-notch and high-quality summaries that highlight essential health information and clinical observations.

It is expected that the combined large language models for clinical text summarization have the potential to transform the delivery of medical services by improving patient experiences and the utilization of resources, especially in health care settings where accuracy is critical. The combination of different models in clinical text summarization provides for both present and potential scalability, making it a viable option for the health sector as medical data becomes substantially more complex and voluminous.

In addition to summarizing medical histories, the ability to extend this proposed approach of combining the 3 large language model for text summarization to other fields highlights its broader applicability and potential to transform practices across diverse fields such as (educational content summarization, technical documentation, news article summarization etc.)

The combination of these models (BART, T5, and Pegasus) into a broad language model ensures that clinical practitioners have access to relevant and comprehensive summaries, ultimately resulting to increase in productivity decisions, more effective and informed practices in the healthcare.

Further research efforts are essential to improve the combined summaries generated from various text summarization models. Assessing their effectiveness across a variety of datasets from various healthcare settings sheds light on how to enhance clinical note summarization and improve patient healthcare information outcome. Also, fine-tuning the model could further enhance its performance, as the synergy of its components often yields superior results compared to individual elements.

8. REFERENCES

Arokodare, O., & Wimmer, H. (2023). Large Language Models for Phishing and Spam Detection: A BERT Approach. *1st Annual Fall National Conference on Creativity, Innovation, and Technology Conference (NCCIT)*, USA.

Batra, P., Chaudhary, S., Bhatt, K., Varshney, S., & Verma, S. (2020). A Review: Abstractive Text Summarization Techniques using NLP. 2020 *International Conference on Advances in Computing, Communication & Materials (ICACCM)*, Dehradun, India, 2020, pp. 23-28, doi: 10.1109/ICACCM50413.2020.9213079.

Bhatia, N., & Jaiswal, A. (2015). Trends in extractive and abstractive techniques in text summarization. *International Journal of Computer Applications*, 117(6), 21-24. DOI:10.5120/20559-2947.

Bijal, D., Nikita, P., & Sanket, S. (2017). A review paper on text summarization for Indian languages. *IJSRD-International Journal for Scientific Research & Development*, 5(7), 744-745.

Delangue, C. (2016). Hugging face. https://huggingface.co/docs/transformers/model_doc/pegasus.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://doi.org/10.48550/arXiv.1810.04805.

Gaikwad, D. K., & Mahender, C. N. (2016). A review paper on text summarization. *International Journal of Advanced Research in Computer and Communication Engineering*, 5(3), 154-160. DOI:10.17148/IJARCCE.2016.5340.

Hanna, M., & Bojar, O. (2021). A fine-grained analysis of BERTScore. In *Proceedings of the Sixth Conference on Machine Translation*, pages 507–517, Online. Association for Computational Linguistics.

HealthIT.gov. (2021). Challenges to Electronic Public Health Reporting among Non-Federal Acute Care Hospitals During the COVID-19 Pandemic. https://www.healthit.gov/data/data-briefs/electronic-public-health-reporting-among-non-federal-acute-care-hospitals-during.

Johnson, A., Pollard, T., Horng, S., Celi, L., & Mark, R. (2023). MIMIC-IV-Note: Deidentified free-text clinical notes (version 2.2). PhysioNet. https://doi.org/10.13026/1n74-ne17.

Johnson, A. E., Stone, D. J., Celi, L. A., & Pollard, T. J. (2018). The MIMIC Code Repository: enabling reproducibility in critical care research. *Journal of the*

American Medical Informatics Association, 25(1), 32-39. DOI: 10.1093/jamia/ocx084.

Kryściński, W., Keskar, N. S., McCann, B., Xiong, C., & Socher, R. (2019). *Neural text summarization: A critical evaluation*. arXiv preprint arXiv:1908.08960. https://doi.org/10.48550/arXiv.1908.08960.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pages* 311-318. https://doi.org/10.3115/1073083.1073135.

Roberts, A., Raffel, C., Lee, K., Matena, M., Shazeer, N., Liu, P. J., Narang, S., Li, W., & Zhou, Y. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. Google, Tech. Rep. https://doi.org/10.48550/arXiv.1910.10683.

Tsai, H.-Y., Huang, H.-H., Chang, C.-J., Tsai, J.-S., & Chen, H.-H. (2022). Patient History Summarization on Outpatient Conversation. 2022 IEEE/WIC/ACM *International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, Niagara Falls, ON, Canada, 2022, pp. 364-370, doi: 10.1109/WI-IAT55865.2022.00060.

Van Veen, D., Van Uden, C., Blankemeier, L., Delbrouck, J.-B., Aali, A., Bluethgen, C., Pareek, A., Polacin, M., Reis, E. P., & Seehofnerova, A. (2023). Clinical text summarization: Adapting large language models can outperform human experts. *Research Square*. doi: 10.21203/rs.3.rs-3483777/v1.

Yang, X., Chen, A., PourNejatian, N., Shin, H. C., Smith, K. E., Parisien, C., Compas, C., Martin, C., Costa, A. B., & Flores, M. G. (2022). A large language model for electronic health records. *NPJ Digital Medicine*, 5(1), 194. https://doi.org/10.1038/s41746-022-00742-2.

Zhang, H., Xu, J., & Wang, J. (2019). Pretraining-based natural language generation for text summarization. arXiv preprint arXiv:1902.09243. https://doi.org/10.48550/arXiv.1902.09243.

Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., & Artzi, Y. (2019). *BERTScore: Evaluating Text Generation with BERT*. ArXiv, abs/1904.09675. https://doi.org/10.48550/arXiv.1904.09675.

APPENDIX

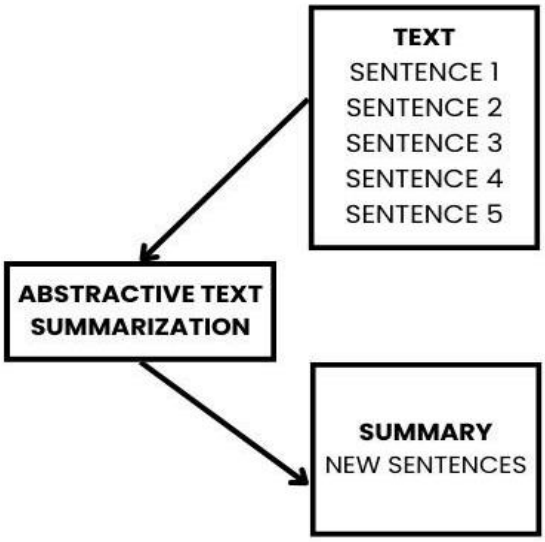

Figure 1 Abstractive Text Summarization Workflow

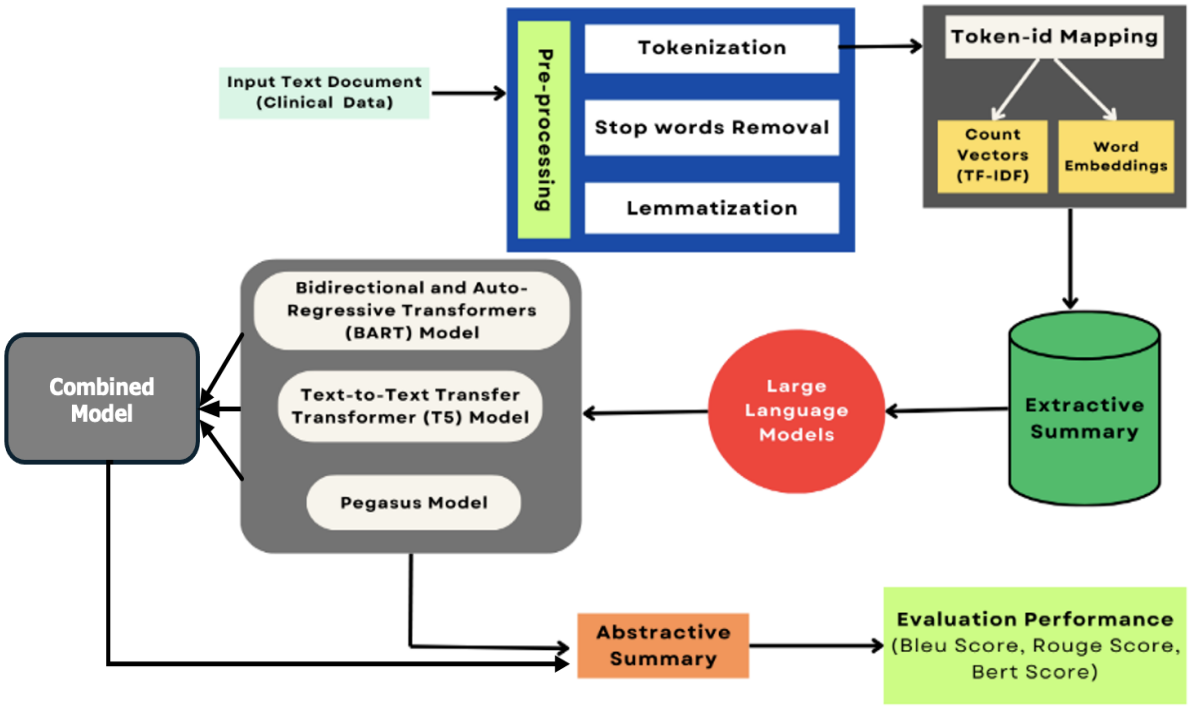

Figure 2 Architecture Diagram for Clinical Text Summarization with Large Language Model

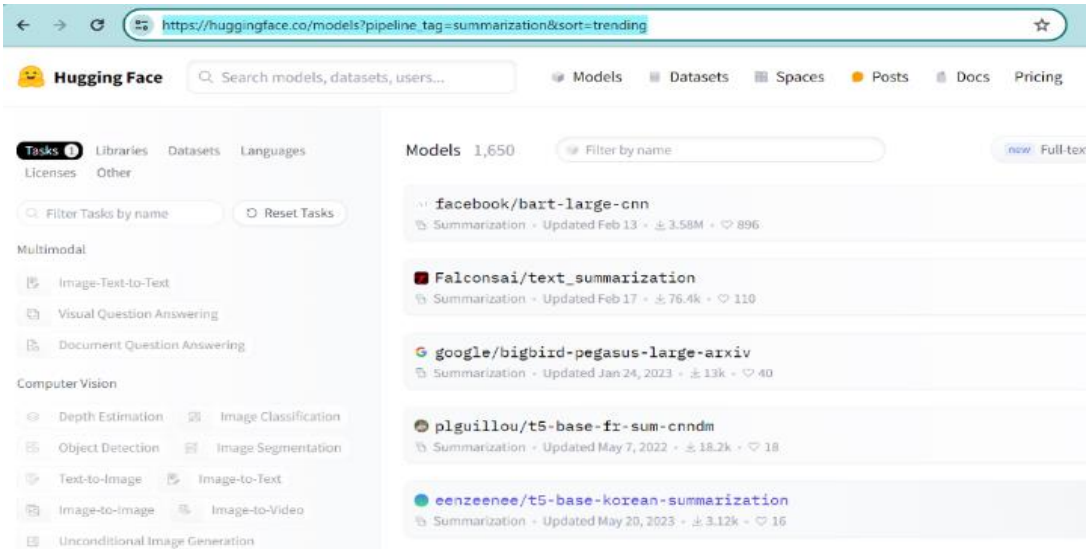

Figure 3 Hugging Face Model Parameters Snapshot for Text Summarization

		Clinical Note 1				
		Scores				
		BLEU	Rouge1	Rouge2	RougeL	BERT(Precision/Recall/F1)
Model	T5	0.002011	0.247191	0.241509	0.247191	0.926807 ,0.813933, 0.866710
	Bart	0.012046	0.304348	0.299270	0.304348	0.918589,0.850790, 0.883391
	Pegasus	0.000545	0.193050	0.186770	0.193050	0.949601,0.812216,0.875552
	Combined Model	0.197573	0.473054	0.445783	0.375449	0.963512,0.838817, 0.896851

Table 1: Clinical Note Test Sample 1

		Clinical Note 2				
		Scores				
		BLEU	Rouge1	Rouge2	RougeL	BERT(Precision/Recall/F1)
Model	T5	0.011963	0.324706	0.307329	0.320000	0.947187, 0.821805, 0.880052
	Bart	0.024612	0.366133	0.358621	0.361556	0.946826,0.826174, 0.882395
	Pegasus	0.062197	0.421286	0.405345	0.421286	0.956568,0.835085, 0.881708
	Combined Model	0.362768	0.575707	0.467446	0.465890	0.962171, 0.846523, 0.892728

Table 2: Clinical Note Test Sample 2

		Clinical Note 3				
		Scores				
		BLEU	Rouge1	Rouge2	RougeL	BERT(Precision/Recall/F1)
Model	T5	0.000002	0.131261	0.117851	0.131261	0.920835,0.800128,0.856248
	Bart	0.000007	0.147260	0.140893	0.113014	0.932428,0.803353,0.863091
	Pegasus	0.000506	0.202322	0.189684	0.202322	0.909739,0.817642,0.861235
	Combined Model	0.058681	0.415205	0.384164	0.339181	0.941585,0.863127,0.900651

Table 3: Clinical Note Test Sample 3

Actual Medical Health History- Case 1	Generated Summary by the Model
"HCV cirrhosis c/b ascites, hiv on ART, h/o IVDU, COPD, bioplar, PTSD, presented from OSH ED with worsening abd distension over past week. Pt reports self-discontinuing lasix and spirnolactone weeks ago, because she feels like they dont do anything and that she doesnt want to put more chemicals in her. She does not follow Na-restricted diets. In the past week, she notes that she has been having worsening abd distension and discomfort. She denies edema, or SOB, or orthopnea. She denies f/c/n/v, d/c, dysuria. She had food poisoning a week ago from eating stale cake n/v 20 min after food ingestion, which resolved the same day. She denies other recent illness or sick contacts. She notes that she has been noticing gum bleeding while brushing her teeth in recent weeks. she denies easy bruising, melena, BRBPR, hemetesis, hemoptysis, or hematuria.Because of her abd pain, she went to OSH ED and was transferred to for further care. Per ED report, pt has brief period of confusion - she did not recall the ultrasound or bloodwork at osh. She denies recent drug use or alcohol use. She denies feeling confused, but reports that she is forgetful at times. In the ED, initial vitals were 98.4 70 106/63 16 97%RA Labs notable for ALT/AST/AP , Tbili1.6, WBC 5K, platelet 77, INR 1.6"	**BART Model** "HCV cirrhosis c/b ascites, hiv on ART, h/o IVDU, COPD, bioplar, PTSD, presented from OSH ED with worsening abd distension over past week. Pt reports self-discontinuing lasix and spirnolactone weeks ago, because she feels like they dont do anything." **T5 Model** "in the past week, she has been having worsening abd distension and discomfort. she denies edema, or SOB, or orthopnea. she had food poisoning a week ago from eating stale cake n/v." **Pegasus Model** "HCV cirrhosis c/b ascites, hiv on ART, h/o IVDU, COPD, bioplar, PTSD, presented from OSH ED with worsening abd distension over past week" **Combined Model (BART, T5 and Pegasus Model)** "Pt with HCV cirrhosis, HIV on ART, COPD, bipolar disorder, and PTSD presented with worsening abdominal distension, discontinue taking lasix and spironolactone because she feels like they dont do anything. in the past week, she has worsened abdominal discomfort, denies edema, SOB, or orthopnea. she had food poisoning from eating stale cake."

Table 4: Comparison between Clinical Text Sample 1 and Summaries Generated by the Four Models

Actual Medical Health History- Case 2	Generated Summary by the Model
"This is a woman with a history of ETOH abuse who presents with abdominal distention, back pain, fever, and elevated white count from Liver Clinic. Ms. was recently admitted to this hospital about 1 week ago for treatment of ascites and work-up of alcoholic hepatitis. At that time she had a diagnostic and therapeutic paracentesis and was treated for a UTI. She wasdischarged home and instructed to follow-up in Liver Clinic in 1 week. On day of presentation to liver clinic, patient complained of worsening abdominal pain and low-grade fevers at home. Her labwork was also significant for an elevated white count. As such, Ms. was admitted for work-up of fever and white count, and for treatment of recurrent ascites. with recently diagnosed alcoholic hepatitis, persistent ascites, and persistent fevers and leukocytosis which have been atributed to her hepatitis who presented to today with worsening abdominal distention, pain, and persistent fever. She denies chills but did have sweats the night prior to admission. She has tried to be strictly compliant with her low socium diet and fluid restriction, and denies any increased fluid or sodium intake. She reports sobriety from alcohol since . At she was febrile and tender to palpation, so she was referred to the ED.In the ED initial vital signs were 99.0 113/72 132 16 99% on RA. Her temp increased to 100.4 and her pulse came down to the 100s with Ativan. She received morphine 4mg IV x 4 for pain, tylenol PO x1 for fever, ondansetron 4mg IV x2 for nausea, and lorazeman 0.5mg IV x1 for anxiety. She underwent a diagnostic paracentesis but the samples were initially lost. She was treated with ceftriaxone 2g IV x1 for possible SBP. She was admitted to Medicine for further management. Fortunately, her samples were found after she arrived on the floor. On the floor her mood is labile. She is at times tearful and at times pleasant. She does seem uncomfortable. She is not confused or obviously encephalopathic. She denies cough, dysuria, diarrhea, or rash. She does endorse decreased UOP for the past few days."	**BART Model** "Ms. was recently admitted to this hospital about 1 week ago for treatment of ascites and work-up of alcoholic hepatitis. At that time she had a diagnostic and therapeutic paracentesis and was treated for a UTI. She was discharged home and instructed to follow-up in Liver Clinic in 1 week. On day of presentation to liver clinic, patient complained of worsening abdominal pain and low-grade fevers at home. Her labwork was also significant for an elevated white count." **T5 Model** "a woman with a history of ETOH abuse presents with abdominal distention, back pain, fever, and elevated white count. she was recently admitted to this hospital about 1 week ago for treatment of ascites and work-up of alcoholic hepatitis. on the day of presentation to liver clinic, patient complained of worsening abdominal pain and low-grade fevers at home. her labwork was also significant for an elevated white count." **Pegasus Model** "This is a woman with a history of ETOH abuse who presents with abdominal distention, back pain, fever, and elevated white count from Liver Clinic. was recently admitted to this hospital about 1 week ago for treatment of ascites and work-up of alcoholic hepatitis. On day of presentation to liver clinic, patient complained of worsening abdominal pain and low-grade fevers at home. with recently diagnosed alcoholic hepatitis, persistent ascites, and persistent fevers and leukocytosis which have been atributed to her hepatitis who presented to today with worsening abdominal distention, pain, and persistent fever." **Combined Model (BART, T5 and Pegasus Model)** "a woman with a history of ETOH abuse presents with abdominal distention, back pain, fever, and elevated white count. She was recently admitted to this hospital about 1 week ago for ascites treatment and alcoholic hepatitis work-up. On presentation day at the liver clinic, she reported worsening abdominal pain and low-grade fevers at home, alongside elevated white count in lab work results".

Table 5: Comparison between Clinical Text Sample 2 and Summaries Generated by the Four Models

Actual Medical Health History- Case 3	Generated Summary by the Model
"yo female with history of Afib on Xarelto, COPD, HTN, PAD who presents for abnormal labs. She noted dark, tarry, stool on and presented to PCP , where H/H was noted to be 8.8/28.6 from prior 11..6 baseline Hct about 38. She has also been experiencing bright red blood with wiping, she believes from her hemorrhoids. PCP called pt who agreed to come to ED. She had colonoscopy in with showed a benign polyp, internal hemorrhoids, and diverticulosis. Her last BM was , was reportedly regular. She currently complains of increased exertional fatigue and has been feeling more SOB than her baseline. Over the last 6 months she has noticed she becomes increasingly out of breath, walking or climbing stairs. She becomes SOB after 6 stairs or less than 1 block, requiring her to stop, and at times use albuterol inhaler. She used to use her only use her inhaler times per day, now she uses it over four times a day and nebulizers twice a day. F with pmhx of COPD nighttime O2, htn, afib who presents with dyspnea, currently being treated for COPD and admitted for Afib with RVR. The patient went to the ED on and was diagnosed with a COPD flare. She was discharged with a prednisone taper currently on 60mg and azithromycin. This AM she initially felt well, then developed dyspnea at rest, worsening with exertion. Her inhalers improved her SOB. She felt that these symptoms were consistent with her COPD. She saw her PCP today in clinic where she was found to be in Afib w/ RVR, rate around 110-120. She has a history of afib. He referred her to the ED for persistent SOB and afib with RVR. She states she been compliant with nebs and steroid/azithro regimen. She denies any edema, orthopnea. She denies recent travel, surgeries. She had an episode of chest tightness this AM that felt like her COPD flares. Denies fevers or coughing or production of sputum, hemomptysis. year old female with history of COPD on home O2, HTN, Afib admitted with dyspnea and cough. Pt states inc dyspnea since this am, also one episode of retrosternal chest pressure lasting 2minuts on way to ED. No cp currently. on home O2. no fevers/chills or abd sx. Patient was recently admitted from with COPD flare and afib with RVR. She could not receive azithromycin due to concern for QTc prolongation and so was treated with ceftriaxone/cefpodoxime. She was treated with 60mg PO prednisone and discharged with a prednisone taper of 10 mg decrease q3d until at 10 mg, then stay at 10 mg until pulm follow up. She was also counseled to do pulmonary rehab and follow up with Dr. . She was discharged on 2L supplemental O2 to be worn at all times. He theophylline was decreased from 300 mg BID to mg BID due to her afib with RVR.She was also seen in the ED on and due to dyspnea which was felt to be a continuation of her COPD flare in the setting of patient not taking her home medications. She was given nebulizers and improved. She was DCed home with for assistance with medications. She declined pulmonary rehab facility disposition."	**BART Model** "The patient went to the ED on and was diagnosed with a COPD flare. She had colonoscopy in with showed a benign polyp, internal hemorrhoids, and diverticulosis. She currently complains of increased exertional fatigue and has been feeling more SOB than her baseline." **T5 Model** "yo female with history of afib on Xarelto, COPD, HTN, PAD. she has been experiencing bright red blood with wiping, believes from her hemorrhoids. over the last 6 months she has noticed she becomes increasingly out of breath." **Pegasus Model** "She became SOB after 6 stairs or less than 1 block, requiring her to stop, and at times use albuterol inhaler. F with pmhx of COPD nighttime O2, htn, afib who presents with dyspnea, currently being treated for COPD and admitted for Afib with RVR. year old female with history of COPD on home O2, HTN, Afib admitted with dyspnea and cough." **Combined Model (BART, T5 and Pegasus Model)** "yo female with history of afib on Xarelto, COPD, HTN, PAD. she has been experiencing bright red blood with wiping, believes from her hemorrhoids. over the last 6 months she has noticed she becomes increasingly out of breath.The patient went to the ED on and was diagnosed with a COPD flare. She had colonoscopy in with showed a benign polyp, internal hemorrhoids, and diverticulosis. She currently complains of increased exertional fatigue and has been feeling more SOB than her baseline.She became SOB after 6 stairs or less than 1 block, requiring her to stop, and at times use albuterol inhaler. F with pmhx of COPD nighttime O2, htn, afib who presents with dyspnea, currently being treated for COPD and admitted for Afib with RVR. year old female with history of COPD on home O2, HTN, Afib admitted with dyspnea and cough."

Table 6: Comparison between Clinical Text Sample 3 and Summaries Generated by the Four Models

Navigating the AI Landscape: An Analysis of AI Developer Tools Usage, Sentiment, and Trust Among IT Professionals

Wendy Ceccucci
wendy.ceccucci@qu.edu
Business Analytics and Information Systems Department
Quinnipiac University
Hamden, CT 06518 USA

Alan Peslak
arp14@psu.edu
Department of Information Sciences and Technology
Penn State University
Dunmore, PA 18512 USA

Kiku Jones
kiku.jones@qu.edu
Business Analytics and Information Systems Department
Quinnipiac University
Hamden, CT 06518 USA

Abstract

GitHub Copilot, developed by GitHub, is a new tool aiding developers with a range of tasks, including the generation of code snippets, documentation assistance, and the formulation of implementation strategies. Comparable AI development tools, such as Tabnine and AWS Code Whisperer, also serve as developmental aids but are utilized to varying and much lesser extents. Our research examines the adoption of GitHub Copilot and similar AI development tools among professional developers and other users using the Stack Overflow Annual Survey. The study reveals notable age-related disparities in tool usage, with younger individuals showing a markedly higher propensity to employ these technologies compared to older users. Additional significant insights include variations in usage based on the type of developer, professional status, and the influence of user attitudes towards AI on the adoption of GitHub Copilot among developers.

Keywords: AI Developer Tools, Artificial Intelligence, GitHub Copilot, Technology Adoption, AI Trust

Recommended Citation: Ceccucci, W., Peslak, A., Jones, K. (2025). Navigating the AI Landscape: An Analysis of AI Developer Tools Usage, Sentiment, and Trust Among IT Professionals. *Journal of Information Systems Applied Research and Analytics*. v18, n1 pp 32-41. DOI# https://doi.org/10.62273/OEMC9700

Navigating the AI Landscape: An Analysis of AI Developer Tools Usage, Sentiment, and Trust Among IT Professionals

Wendy Ceccucci, Alan Peslak and Kiku Jones

1. INTRODUCTION

The integration of Artificial Intelligence (AI) in software development has marked a transformative era in the information technology (IT) sector. AI developer tools, ranging from automated code assistants to advanced debugging algorithms, have become pivotal in shaping how IT professionals approach software development. This paper presents a comprehensive analysis of the usage patterns, sentiment, and trust levels associated with AI developer tools among IT developers. The study aims to provide an understanding of how these tools are reshaping the development landscape, the attitudes of developers towards them, and the trust issues that may arise.

The advent of AI in development tools has introduced both opportunities and challenges. While these tools promise increased efficiency, accuracy, and even creativity in coding, they also raise questions about reliability, ethical use, and the potential for diminishing human skill sets. Understanding the sentiments and trust levels of developers towards AI tools is crucial for the future design, implementation, and governance of these technologies. According to a developer survey by GitHub (2024) 92% of developers working in large companies are already using AI tools either at work or in their personal time.

This paper begins by exploring the current landscape of AI developer tools, categorizing them based on their functionalities and the aspects of software development they impact. We then delve into the methodologies used to gauge the usage patterns of these tools among IT professionals. This includes a survey of a diverse group of developers, encompassing various industries, experience levels, and geographical locations.

Following the usage analysis, the paper addresses the core aspects of sentiment and trust. Sentiment analysis is conducted through survey responses. This analysis provides insights into the general attitudes of developers towards these tools, ranging from enthusiasm and optimism to skepticism and concern. Trust, a more complex and multifaceted issue, is examined through a separate survey response.

The findings of this study are expected to offer valuable insights for multiple stakeholders. Tool developers and AI researchers can gain a better understanding of user attitudes and trust factors, guiding them in creating more user-centric and trustworthy tools. Organizations and team leaders can use these insights to make informed decisions about integrating AI tools into their workflows, considering both the technical and human aspects. Additionally, the study contributes to the broader discourse on the role of AI in the future of work, particularly in the IT sector.

This paper aims to shed light on the complex dynamics between IT developers and AI developer tools, focusing on usage patterns, sentiments, and trust. By providing an analysis of these aspects, the study seeks to contribute to the responsible and effective integration of AI in software development practices.

2. LITERATURE REVIEW

AI Code Generation Tools

Ai code generation systems are simply AI systems trained to write code, thereby automating the process of programming. The code generating system is built upon machine learning algorithms that train on large datasets of code. The algorithms learn the common patterns, practices, and conventions in coding and then can generate new code based on their learning.

AI coding tools are becoming standard practice for many software developers. According to an online survey conducted by Wakefield Research on the behalf of GitHub, 92% of U.S. developers are using AI coding tools both in and outside of work (Shani & GitHub Staff, 2023). The survey responses were from 500 U.S. based developers who were not managers and worked at companies with 1,000-plus employees. They also found that more than 4 out of 5 developers expected AI coding tools would make their team

more collaborative.

The most widely used AI developer tool is GitHub's Copilot which was released for technical preview in 2021. GitHub Copilot is an AI-powered code assistant developed by GitHub in collaboration with OpenAI. GitHub Copilot is designed to help developers write better code faster and with fewer errors. It operates by analyzing the context within the codebase and suggesting whole lines or blocks of code that developers can incorporate or modify as needed.

GitHub Copilot, an AI-powered code assistant, is used by software developers primarily in JavaScript and Python, with Visual Studio Code being the main IDE. It is most commonly used for data processing and code generation, with the potential to significantly reduce development time and slightly increase code quality. However, concerns about security and difficulty of integration have been raised, and developers are divided in their opinions about the tool (Jaworski & Piotrkowski, 2023; Zhang, et al., 2023).

Other examples of AI based code assistants include Tabnine (Kedar, 2024), and CodeWhisperer (Desai & Deo 2022). Comparing the tools:

1. GitHub Copilot is owned by Microsoft and is based on OpenAI's Codex model. One of its strengths is its flexibility in deployment (cloud vs local). It excels in its deep integration with GitHub and its powerful language model but is best suited for those already invested in the GitHub ecosystem.
2. Tabnine based on various models including GPt-3. It also offers flexibility in deployment and a wide range of IDE and language support, which is beneficial for diverse development environments.
3. Amazon CodeWhisperer is developed by Amazon Web Services (AWS). It is based on AWS's proprietary models, including adaptations of large language models. The context-aware code suggestions are similar to the other tools but tailored for cloud and serverless environments.

According to a Survey by CodeSignal (2024) of 1,000 developers worldwide, 81% of developers surveyed said they use AI-powered coding assistants. The number one reason for using the tool was for learning new technical skills or knowledge. The second most common reason was for generating boilerplate code.

Some of the advantages of AI code generation tools include:

1. Saves time. These tools can accelerate the development cycle, making the process more efficient.
2. Learning and skill development. AI code generation supports learning and skill enhancement. By observing the AI-generated code developers can learn new techniques, understand different code structures, and adopt better coding practices.
3. Accessibility for non-experts. Individuals with less coding knowledge can create functional applications by simply describing the requirements in natural language.
4. Reduces human error. AI tools can help minimize human errors and the AI-generated code is often well-documented with explanations making it easier to understand and debug.

Along with its potential advantages, AI code generated tools have some limitations. These include:

1. Quality and accuracy. AI-generated code may not always meet the quality and accuracy standards required for complex projects (Hasselbring & Reussner, 2006; Lipner, 2004).
2. Contextual understanding. AI tools often fail to fully grasp specific project contexts, resulting in misaligned code (Wang, et al., 2024).
3. Dependency on data. The effectiveness of AI code generation tools is highly dependent on the quality and breadth of their training data. Poor or biased data leads to poor performance.
4. Security concerns. AI-generated code may introduce vulnerabilities if secure coding practices are not properly embedded. According to a study by Snyk (2023), more than three-quarters of developers bypass established protocols to use code completion tools despite potential risks.
5. Limited creativity. AI lacks the creativity and problem-solving abilities inherent to human developers, leading to less innovative solutions.
6. Trust and reliability. Developers often find it challenging to trust AI-generated code without extensive testing and

verification, increasing their workload (Wang, et al., 2024).
7. Ethical and legal issues. The use of AI in code generation raises concerns related to intellectual property and the legality of using certain code snippets.

According to a study by Dakhel, et al. (2023), Copilot can be an asset in software projects when used by expert developers, as its suggestions can match human contributions in quality. However, Copilot can become a liability if used by novice developers who may struggle to filter its suggestions effectively.

Trust and AI Tools

Recent studies have explored the impact of AI-powered code generation tools on software developers' trust and the potential for these tools to automate routine programming tasks (Cheng et al., 2024; Ernst & Bavota, 2022). These tools, such as GitHub Copilot, have the potential to increase the level of abstraction in software development, allowing developers to focus on business processes rather than code (Palacios-González, et al., 2008). However, concerns have been raised about the potential for bias, legal compliance, and security vulnerabilities in AI-driven development environments (Ernst & Bavota, 2022).

One significant challenge involves helping users evaluate the trustworthiness of AI tools. The importance of software developers' trust in programming has been extensively studied, highlighting it as a crucial design requirement for these tools. Trust is considered a key prerequisite for ensuring the safety of the resulting software products (Hasselbring & Reussner, 2006; Lipner, 2004).

According to a study by Wang, et al. (2024), developers' trust is rooted in the AI tool's perceived ability, integrity, and benevolence, and is situational, varying according to the context of usage. Existing AI code generation tools lack the affordances for developers to efficiently and effectively evaluate the trustworthiness of AI-powered code generation tools.

Several other demographics and factors can influence a developer's trust in AI tools, and understanding these can help tailor AI solutions to better meet user needs. These factors include developer's geographical location, gender, and socioeconomic status.

Developers from different regions may have varying levels of trust in AI tools based on cultural attitudes towards technology and data privacy. Although their research did not focus on developers specifically, Grassini and Ree (2023) found that respondents from the USA demonstrated higher levels of hopefulness for AI technology compared to those from the UK. Their finding suggests that cultural context does play a role in shaping individuals' perceptions of AI technology.

In the same study, they found that male respondents showed higher hopes for AI systems than female respondents. This finding is consistent with previous research, which reported that males perceive AI as more useful (Arujo, et al., 2020) and generally more favorable than females (Lozano, et al., 2021). Likewise, a study by Armutat, et al. (2024) found that men tended to view AI applications more positively, rate their own AI competencies higher, and have more trust in the technology compared to women.

Age and AI Coding Tools

The integration of AI tools into the workflow of software developers has shown a notable age-related trend, with younger developers demonstrating a higher propensity to adopt such technologies. This trend can be attributed to several factors. First, younger developers are generally more exposed to the latest technological innovations during their education and early careers, making them more receptive to AI-driven solutions. Educational institutions are increasingly incorporating AI and machine learning courses into their curricula, which equips new graduates with the skills and familiarity needed to leverage these tools effectively.

Moreover, the rapid pace of technology means that younger individuals often have a 'digital native' advantage. They tend to be more adaptive to changes in the tech landscape, including the use of sophisticated AI platforms that require a steep learning curve. A report by GitHub, The State of the Octoverse (Daigle & GitHub Staff, 2024), highlights that newer programmers are more likely to utilize AI coding assistants, attributing this to their up-to-date training and inherent flexibility in adopting new workflows.

Additionally, the software development industry itself fosters a culture of innovation and continuous learning, which resonates well with the mindset of younger developers. They are

often driven by the potential to streamline coding processes, enhance productivity, and solve complex problems efficiently with the help of AI tools. In practical terms, younger developers utilize AI tools for a range of tasks including writing and reviewing code, automating repetitive tasks, and even in conceptual phases of development like brainstorming and prototyping. The convenience and efficiency offered by AI tools align with the fast-paced, results-oriented environment preferred by this demographic. The influence of AI on younger developers is not just transforming their individual workflows but is also shaping the future dynamics of team collaborations and project management in software development. As these young developers progress in their careers, their preference for AI-enhanced workflows is likely to catalyze broader adoption of these technologies across the industry, potentially leading to more innovative solutions in the overall approach to software development.

3. METHODOLOGY

This study explores the usage of AI tools by software developers by leveraging data from the 2023 Stack Overflow survey. The Stack Overflow Developer Survey is widely acknowledged as the most comprehensive survey targeting coders worldwide. Annually, it covers a wide range of subjects, from developers' technology preferences to their career aspirations. According to Stack Overflow's website:

> "For 13 years, we've delivered industry-leading insights regarding the developer community. This is the voice of the developer. Analysts, IT leaders, reporters, and other developers turn to this report to stay up to date with the evolving developer experience, technologies that are rising or falling in favor, and to understand where tech might be going next." (Stack Overflow, 2023)

The validity of using Stack Overflow data is supported by its frequent citation in numerous peer-reviewed publications, including studies by Barua, et al. (2014), Asaduzzaman, et al. (2013), and Treude and Robillard (2016). The dataset comprises a rich mix of demographic data, descriptive statistics, and responses to opinion-based questions about the programming industry. IBM SPSS 29 was utilized for the data analysis. Primarily descriptive statistics and crosstab analyses were used in the study.

4. RESULTS

Survey respondents were asked, "Which AI-powered developer tools did you use regularly over the past year and which do you plan to work with over the next year?"

Table 1 displays the results of this survey question for all participants. Out of 89,870 total responses, 686 respondents did not answer the question. This leaves a total of 89,184 valid responses.

AI Developer Tool	Use		Do Not Use	
GitHub Copilot	22,078	24.76%	67,106	75.24%
Tabnine	5,193	5.82%	83,991	94.18%
AWS CodeWhisperer	2,071	2.32%	87,113	97.68%
Synk Code	538	0.60%	88,646	99.40%
Codeium	504	0.57%	88,680	99.43%
Whispr AI	455	0.51%	88,729	99.49%
Rubber Duck.AI	151	0.17%	89,033	99.83%
Mintify	-	0.00%	89,184	100.00%
Replit Ghostwrite	-	0.00%	89,184	100.00%

Table 1: AI- Powered Developer Tool Usage by All Respondents

In the next table, Table 2, the responses were limited to software developers only. A total of 67,973 respondents identified as software developers by profession. With 686 respondents not answering the question, the total number of valid responses was 67,287.

AI Developer Tool	Use		Do Not Use	
GitHub Copilot	17,432	25.93%	49,805	74.07%
Tabnine	3,651	5.43%	63,586	94.57%
AWS CodeWhisperer	1,527	2.27%	65,710	97.73%
Synk Code	413	0.61%	66,824	99.39%
Codeium	333	0.50%	66,904	99.50%
Whispr AI	280	0.42%	66,957	99.58%
Rubber Duck.AI	79	0.12%	67,158	99.88%
Mintify	-	0.00%	67,237	100.00%
Replit Ghostwrite	-	0.00%	67,237	100.00%

Table 2: AI- Powered Developer Tool Usage by Software Developers

The main AI developer tool is clearly GitHub Copilot, with approximately 25% usage by both groups. The next most used tool is Tabnine, at about 5%. Given the dominance of GitHub Copilot, the focus of the rest of our paper will be on this tool.

The next question related to AI development tools usage asked the question: "Do you currently use AI tools in your development process?" The three possible responses were "Yes", "No, but I plan to soon", and "No, I don't

plan to". Table 3 shows the results. Overall, the study reveals that 44% of Developers currently use AI tools and another 26% plan to soon. Only 30% do not have plans to use AI.

Response	Count	Percent
Yes,	29,697	44.20%
No, but I plan to	17,401	25.90%
No, I don't plan to	20,139	30%
Total	67,237	100%

Table 3: Plans to Use AI- Powered Developer Tool by Software Developers

The survey then looked at how respondents viewed AI (Table 4). Overall, 76% of developers have a favorable view of AI. Table 5 shows the effect of attitude towards AI and the use of GitHub Copilot. Those with a very favorable view use Copilot nearly 50%. This rate drops for other views hovering in the 20% range for those with neutral or unfavorable views.

Attitude	Frequency	Percent	Valid Percent
Very Favorable	13,002	19.30%	27.70%
Favorable	22,717	33.80%	48.40%
Indifferent	9,705	14.40%	20.70%
Unfavorable	1,305	1.90%	2.80%
Very Unfavorable	199	0.30%	0.40%
Total	46,928	69.80%	100%
Missing	20,309	30.20%	
Total	67,237	100%	

Table 4: Software Developer's Attitudes Toward AI

		GitHub Copilot		
Attitude	Statistic	Use	Do Not Use	Total
Very Favorable	Count	6,421	6,581	13,002
	Percentage	49.4%	50.6%	
Favorable	Count	7,768	14,949	22,717
	Percentage	34.2%	65.8%	
Indifferent	Count	1,935	7,770	9,705
	Percentage	19.9%	80.1%	
Unfavorable	Count	292	1,013	1,305
	Percentage	22.4%	77.6%	
Very Unfavorable	Count	54	145	199
	Percentage	27.1%	72.9%	
Total	Count	16,470	30,458	46,928
	Percentage	35.1%	64.9%	

Table 5: Software Developer's Attitudes Toward AI and use of GitHub Copilot

The survey then asked respondents "How much do you trust the accuracy of the output from AI tools as part of your development workflow?" Table 6 shows the survey results. Sentiment is higher than trust in the accuracy of AI output with only 2% of developers having high trust. There is a large percentage who somewhat trust AI at 38%. Twenty-eight percent distrust AI and 32% are neutral.

Level of Trust	Frequency	Percent	Valid Percent
Highly Trust	1,139	1.7%	2.4%
Somewhat Trust	17,696	26.3%	37.8%
Neither Trust Nor Distrust	14,789	22.0%	31.6%
Somewhat Distrust	10,514	15.6%	22.4%
Highly Distrust	2,721	4.0%	5.8%
Total	46,859	69.7%	100.0%
Missing	20,378	30.3%	
Total	67,237	100.0%	

Table 6: Software Developer's Attitudes Toward AI and use of GitHub Copilot

The effect of trust on the usage of GitHub Copilot is revealing. Lack of trust negatively affects adoption with 39% of those who at least somewhat trust AI output using GitHub Copilot. But the percentage of developers with neutral or lack of trust only drops to about 32%. This finding, though significant, demonstrates that trust has a limited impact on the use of AI development tools.

		GitHub Copilot		
Level of Trust	Statistic	Use	Do Not Use	Total
Highly Trust	Count	448	691	1,139
	Percentage	39.3%	60.7%	
Somewhat Trust	Count	6,957	10,739	17,696
	Percentage	39.3%	60.70%	
Neither Trust Nor Distrust	Count	4,898	9,891	14,789
	Percentage	33.10%	66.90%	
Somewhat Distrust	Count	3,286	7,228	10,514
	Percentage	31.30%	68.70%	
Highly Distrust	Count	874	1,847	2,721
	Percentage	32.10%	67.90%	
Total	Count	16,463	30,396	46,859
	Percentage	35.10%	64.90%	

Table 7: Software Developer's Trust in AI and use of GitHub Copilot

Age is clearly a factor in the use of GitHub Copilot (Table 8). This result has been supported by the literature. Nearly one half of the developers under 18 use it and the percentage declines for each older age group. The 18-24 users check in at 35%, 25-34 at 27% and so on. The oldest group over 65 only has a 6% participation rate.

Age	Statistic	GitHub Copilot Use	Do Not Use	Total
Under 18 years old	Count	226	196	422
	Percentage	53.6%	46.4%	
18- 24 years old	Count	3,822	7,180	11,002
	Percentage	34.7%	65.3%	
25-34 years old	Count	7,692	21,156	28,848
	Percentage	26.7%	73.3%	
35 - 44 years old	Count	4,172	13,132	17,304
	Percentage	24.1%	75.9%	
45- 54 years old	Count	1,217	5,270	6,487
	Percentage	18.8%	81.2%	
55-64 years old	Count	277	2172	2449
	Percentage	11.3%	88.7%	
65 years or older	Count	38	556	594
	Percentage	6.4%	93.6%	
Prefer Not to Say	Count	18	113	131
	Percentage	13.7%	86.3%	
Total	Count	49,805	17,432	67,237
	Percentage	74.1%	25.9%	

Table 8: Software Developer's Age and use of GitHub Copilot

When we examine all respondents, we find that both hobbyists and developers by profession have the highest use of GitHub Copilot (Table 9). Surprisingly, those learning to code are lower than these groups at 23%.

Developer Level	Statistic	GitHub Copilot Use	Do Not Use	Total
I am a developer by profession	Count	17,432	49,805	67,237
	Percentage	25.9%	74.1%	
I am learning to code	Count	1,161	3,800	4,961
	Percentage	23.4%	76.6%	
I am not primarily a developer, but I write code sometimes as part of my work/studies	Count	1,835	7,119	8,954
	Percentage	20.5%	79.5%	
I code primarily as a hobby	Count	1,318	3,642	4,960
	Percentage	26.6%	73.4%	
I used to be a developer by profession but no longer am	Count	332	1,529	1,861
	Percentage	17.8%	82.2%	
None of these	Count	0	1211	1211
	Percentage	0.0%	100.0%	
Total	Count	22,078	67,106	89,184
	Percentage	24.8%	75.2%	

Table 9: Developer's Level and use of GitHub Copilot

When we examine the developer type and use of GitHub Copilot, we find that Blockchain, Developer Advocates, and Front-end developers have the highest usage rate. They all exceed 30% usage (Appendix 1). Database, enterprise, and embedded developers all are lowest end at less than 15%.

5. CONCLUSIONS

GitHub Copilot was found to be used the most of all AI powered developer tools regardless of a person's profession. When looking strictly at software developers, the same result was found with GitHub Copilot being used by 25% of the respondents and the next tool coming in at only 5%. An additional finding to help tool developers is that a majority of developers have a favorable view of AI and somewhat trust it. The favorable view of AI does seem to impact a developer's choice in using the AI developer tool. However, trust in AI appears to have limited impact. This is an area where future researchers will want to look further to determine what other factors are possibly moderating this relationship.

Age has a clear impact on the use of AI developer tools. As the age group increased, the use of AI developer tools decreased. This finding, which is supported in the literature, may be helpful to organizations as they review their workforce and determine potential training or motivating opportunities to engage different age groups to participate in using the AI developer tools. Another finding which can be helpful to organizations is that those learning to code do not use the AI developer tools as much as professional developers or even hobbyists. Showing the benefits of learning to code utilizing the AI developer tools and the time saving in completing projects will likely be a positive for both the organization and the developer. If there are developers on site who are already proficient in using the AI developer tools, it may be beneficial to the organization to have these experienced developers work with those who are just beginning to discover the AI developer tools. In addition, those developer types who have the highest usage rate of AI developer tools may be the best people to talk to the other developers who are not utilizing the tools to discuss their experiences to provide credibility to the recommendation to use the tools.

6. REFERENCES

Araujo, T., Helberger, N., Kruikemeier, S. & Vreese, C. (2020). In AI we trust? Perceptions about automated decision-making by artificial intelligence. *AI & Society*, 35, 611-623. https://doi.org/10.1007/s00146-019-00931-w

Armutat, S., Wattenber, M., & Mauritz, N. (2024). Artificial Intelligence – Gender-Specific Differences in Perception, Understanding, and Training Interest. *Proceedings of the 7th International Conference on Gender Research,* 7(1), 36-43. https://doi.org/10.34190/icgr.7.1.2163

Asaduzzaman, M., Mashiyat, A. S., Roy, C., &

Schneider, K. (2013). Answering questions about unanswered questions of stack overflow. *Proceedings of the 10th Working Conference on Mining Software Repositories,* 97-100. https://doi.org/10.1109/MSR.2013.6624015

Barua, A., Thomas, S., & Hassan, A. (2014). What are developers talking about? An analysis of topics and trends in stack overflow. *Empirical Software Engineering*, *19*, 619-654. https://doi.org/10.1007/s10664-012-9231-y

Cheng, R., Wang, R., Zimmermann, T., & Ford, D. (2024). "It would work for me too": How Online Communities Shape Software Developers' Trust in AI-Powered Code Generation Tools. *ACM Transactions on Interactive Intelligent System*s. 14(2), 1–39. https://doi.org/10.1145/3651990

CodeSignal (2024). *Trends Report Developers & AI Coding Assistant Trends*. CodeSignal. Retrieved July 26, 2024, from https://codesignal.com/report-developers-and-ai-coding-assistant-trends/

Daigle, K. & GitHub Staff (2024). *The State of the Octoverse.* GitHub. Retrieved on July 26, 2024, from https://GitHub.blog/news-insights/research/the-state-of-open-source-and-ai/

Desai, A. & Deo, A. (2022*). Introducing Amazon CodeWhisperer, the ML-powered coding companion*. AWS. Retrieved July 26, 2024, from https://aws.amazon.com/blogs/machine-learning/introducing-amazon-codewhisperer-the-ml-powered-coding-companion/

Dakhel, A., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M., & Jiang, Z. (2023). GitHub Copilot AI pair programmer: Asset or Liability?, *Journal of Systems and Software*, 203. https://doi.org/10.1016/j.jss.2023.111734

Ernst, N., & Bavota, G. (2022). AI-Driven Development Is Here: Should You Worry? *IEEE Software*, 39(2) 106-110. https://doi.org/10.48550/arXiv.2204.07560

GitHub (2024). *GitHub Developer Survey*. GitHub. Retrieved July 26, 2024, from https://GitHub.blog/news-insights/research/survey-reveals-ais-impact-on-the-developer-experience/#developers-want-more-opportunities-to-upskill-and-drive-impact

Grassini, S. & Ree, A. (2023). Hope or Doom AI-ttitude? Examining the Impact of Gender, Age, and Cultural Differences on the Envisioned Future Impact of Artificial Intelligence on Humankind. *Proceedings of the European Conference on Cognitive Ergonomics 2023,* 1-7. https://doi.org/10.1145/3605655.3605669

Hasselbring, W. & Reussner, R. (2006). Toward trustworthy software systems. *Computer*, 39(4), 91–92. https://doi.org/10.1109/MC.2006.142

Jaworski, M., & Piotrkowski, D. (2023). Study of software developers' experience using the GitHub Copilot Tool in the software development process. ArXiv, abs/2301.04991. https://doi.org/10.48550/arXiv.2301.04991

Kedar, S. (2024). *Tabnine vs. GitHub Copilot.* Tabnine. Retrieved on July 26, 2024, from https://www.tabnine.com/blog/tabnine-versus-GitHub-copilot/

Lipner, S. (2004). The trustworthy computing security development lifecycle. *Proceedings of the 20th Annual Computer Security Applications Conference,* 2–13. https://doi.org/10.1109/CSAC.2004.41

Lozano, I., Molina, J. & Gijón, C. (2021). Perception of artificial intelligence in Spain. *Telematics and Informatics*, 63. https://doi.org/10.1016/j.tele.2021.101672

Palacios-Gonzalez, E., Fernandez-Fernandez, H., Diaz, V., Garcia-Bustelo, B., & Lovelle, J. (2008). A review of Intelligent Software Development Tools*. Proceedings of the 2008 International Conference on Artificial Intelligence*, 585-590.

Shani, I., & GitHub Staff (2023). *Survey reveals AI's impact on the developer experience.* GitHub. Retrieved July 26, 2024, from https://GitHub.blog/2023-06-13-survey-reveals-ais-impact-on-the-developer-experience/

Snyk (2023). *2023 Snyk AI-Generated Code Security Report.* Snyk. Retrieved on July 26, 2024, from https://snyk.io/reports/ai-code-security/

Stack Overflow Retrieved July 26, from https://survey.stackoverflow.co/2020#overview

Treude, C., & Robillard, M. P. (2016). Augmenting API documentation with insights from stack overflow. *Proceedings of the 38th International Conference on Software Engineering,* 392-403. https://doi.org/10.1145/2884781.2884800

Wang, R., Cheng, R., Ford, D., & Zimmerman, T. (2024). Investigating and Designing for Trust in AI-powered Code Generation Tools. *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency, 1475-1493.* https://doi.org/10.1145/3630106.3658984

Zhang, B., Liang, P., Zhou, X., Ahmad, A., & Waseem, M. (2023). Demystifying Practices, Challenges and Expected Features of Using GitHub Copilot. *International Journal of Software Engineering and Knowledge Engineering*, 33(1) 1653-1672. https://doi.org/10.1142/S0218194023410048

Appendix 1

Developer Type	Statistic	GitHub Copilot		
		Use	Do Not Use	Total
Academic Researcher	Count	294	1,060	1,354
	Percentage	**21.7%**	78.3%	
Blockchain	Count	122	201	323
	Percentage	37.8%	62.2%	
Cloud Infrastructure Engineer	Count	275	761	1,036
	Percentage	**26.5%**	73.5%	
Data or Business Analyst	Count	118	719	837
	Percentage	**14.1%**	85.9%	
Data Scientist or Machine Learning Specialist	Count	441	1,147	1,588
	Percentage	**27.8%**	72.2%	
Database Administrator	Count	34	223	257
	Percentage	**13.2%**	86.8%	
Designer	Count	44	237	281
	Percentage	**15.7%**	84.3%	
Developer Advocate	Count	74	138	212
	Percentage	**34.9%**	65.1%	
Developer Experience	Count	85	241	326
	Percentage	**26.1%**	73.9%	
Developer, Back-End	Count	3,123	10,622	13,745
	Percentage	**22.7%**	77.3%	
Developer, Desktop or Enterprise Apps. or Devices	Count	483	3,421	3,904
	Percentage	**12.4%**	87.6%	
Developer, Embedded Apps. or Devices	Count	215	1,630	1,845
	Percentage	**11.7%**	88.3%	
Developer, Front-End	Count	1,573	3,498	5,071
	Percentage	**31.0%**	69.0%	
Developer, Full-Stack	Count	7,582	18,153	25,735
	Percentage	**29.5%**	70.5%	
Developer, Game or Graphics	Count	161	705	866
	Percentage	**18.6%**	81.4%	
Developer, Mobile	Count	536	2,061	2,597
	Percentage	**20.6%**	79.4%	
Developer, QA or Test	Count	91	495	586
	Percentage	**15.5%**	84.5%	
DevOps Specialist	Count	355	1,032	1,387
	Percentage	**25.6%**	74.4%	

Developer Type	Statistic	GitHub Copilot		
		Use	Do Not Use	Total
Educator	Count	91	324	415
	Percentage	**21.9%**	78.1%	
Engineer, Data	Count	261	987	1,248
	Percentage	**20.9%**	79.1%	
Engineer, Site Reliability	Count	109	318	427
	Percentage	**25.5%**	74.5%	
Engineering Manager	Count	562	1,471	2,033
	Percentage	**27.6%**	72.4%	
Hardware Engineer	Count	29	257	286
	Percentage	**10.1%**	89.9%	
Marketing or Sales Professional	Count	25	124	149
	Percentage	**16.8%**	83.2%	
NA	Count	3,042	9,270	12,312
	Percentage	**24.7%**	75.3%	
Other	Count	585	2,495	3,080
	Percentage	**19.0%**	81.0%	
Product Manager	Count	89	357	446
	Percentage	**20.0%**	80.0%	
Project Manager	Count	91	498	589
	Percentage	**15.4%**	84.6%	
Research & Development Role	Count	254	1,099	1,353
	Percentage	**18.8%**	81.2%	
Scientist	Count	51	300	351
	Percentage	**14.5%**	85.5%	
Security Professional	Count	112	362	474
	Percentage	**23.6%**	76.4%	
Senior Executive	Count	486	846	1,332
	Percentage	**36.5%**	63.5%	
Student	Count	580	1,416	1,996
	Percentage	**29.1%**	70.9%	
System Administrator	Count	105	638	743
	Percentage	**14.1%**	85.9%	
Total	Count	22,078	67,106	89,184
	Percentage	**24.8%**	75.2%	

Applying Design Science to RPA and AI-Based Systems

Biswadip Ghosh
bghosh@msudenver.edu
Computer Information Systems and Business Analytics
Metropolitan State Univerisity of Denver
Denver, Colorado 80217, USA.

Abstract

The organizational adoption and use of computer systems incorporating artificial intelligence (AI) or robotic process automation (RPA) is increasing. The goals are to streamline business processes and improve their efficiency and effectiveness. However, the adoption and use of these AI technologies can manifest complications in human/system interfaces in diverse parts of the organization. Design science research (DSR) emphasizes the creation of innovative artifacts and computer solutions, keeping user goals at the forefront, and has the potential to avert such downstream system issues. Successful systems must be designed to easily coexist with humans and support the collaboration between human and machine actors. This research study investigates the impact of applying design science methodologies in the implementation of automated systems that incorporate AI or RPA. The interview data is collected and analyzed from an agricultural dairy farm automation case study. The results support the benefits of using DSR methodology and are applicable to any AI-based system design/implementation with human components.

Keywords: Design Science, Systems Analysis, Artificial Intelligence, Robotic Process Automation.

Recommended Citation: Ghosh, B., (2025). Applying Design Science to RPA and AI-Based Systems. *Journal of Information Systems Applied Research and Analytics* v18, n1 pp 42-50. DOI# https://doi.org/10.62273/ORIT3319.

Applying Design Science to RPA and AI-Based Systems

Biswadip Ghosh

1. INTRODUCTION

The organizational adoption and use of business systems incorporating functionality based on artificial intelligence (AI) and robotic process automation (RPA) is growing. These technologies provide an opportunity to streamline processes and improve efficiency and effectiveness in various industries such as manufacturing, logistics, transportation, defense, and agriculture. RPA is a lightweight automation technology being applied to automation of high volume, routine, and repetitive work, and is particularly well suited to monitoring status coming from control systems, system-to-system integration events and user interface signals. AI is a more sophisticated technology that is applied to more complex scenarios and less well-defined work tasks. In contrast, RPA connects events with automated actions based on conditional statements and is a key interface technology for repetitive responses to routine external triggers.

Studies show that interfaces that connect AI-based systems to human users must provide rapid operational context, transparency, and explain-ability to help the human user better understand the autonomous system's decision making and behavior in real time and adjust as needed (Azafrani & Gupta, 2023). For more complex automation scenarios, popular AI models apply rule-based or case-based reasoning is paired with human judgement to make decisions and execute actions. For example, in autonomous weapons systems, RPA and AI technology is being used together with human decision making to enhance military-civilian interfaces (Froding & Paterson, 2021). In such weapons applications, the human element provides the balance between the need to mitigate the potential for societal harm and the effectiveness of the military mission.

The introduction of RPA and AI vastly changes the roles played by human actors in the workplace. There is greater risk that the use of AI and RPA can result in organizational issues downstream (Staaby et.al., 2021). The typical implementation focus of these technologies is primarily localized optimization, and downstream issues can manifest in other parts of the organization. These issues include automation bias, unforeseen system events, unexpected errors, overdependence on technology causing obsolescence and deskilling of human actors.

Such ramifications increase the need to adopt design science (DSR) methodologies so that automation systems can be designed/implemented to easily coexists with humans and support the collaboration between the two actors – human and machine. Though there is greater interest in the use of RPA and AI technologies in organizations, there is still limited research studies on how process automation and artificial intelligence applications can be best integrated responsibly into business processes that depend heavily on human creativity and input. The systems' design methodology must view the technology and the human actors as a system of systems – a hybrid system - and build on the synergies of interplay of all actors to improve overall outcomes. For example, in a military RPA interface, background data can be collected by the weapons software agent and decisions suggested in rank order to the human to take final action steps to trigger the weapon (Vassilakopoulou et.al., 2023).

Design Science research (DSR) stems from the evaluation of a system from a user-centric lens. The design science methodology for realizing system artifacts consists of iterative implementations, and comprehensive metrics for usage together with measurements for benefit realization. As new system artifacts are planned and/or built, design science research evaluates these artifacts for their use and value and generates possible explanations for changes in the behavior of systems, people, and organizations (Vaishnavi & Kuechler, 2004). This new approach is a response to the increasing complexity of modern technology and modern business and applies the principles of design to specify systems to relate to the way people work. Ideally, design science can be applied to establish common system goals between both actors – automation and human - via interfaces that support transparency, reciprocity, and sustainable interactions (Venable et.al., 2016).

Research Goals

The goal of this research is to apply design science methodologies to achieve successful implementation of automated systems, and to evaluate any resulting benefits or failings of such methodology. The research premise is that design science evaluation of automation-human system makes long-term human-machine collaboration more effective and eliminate detrimental downstream issues in these systems. Systems success results from managing functional needs to exploit a mix of human, and automation resources to reduce complexity and uncertainty in business processes, and support a balance between all the actors. Successful systems implement strict division of labor, sustain organizational norms, create cost efficiency, and manage all stakeholders in effective roles (Gottschalk & Solli-Saether, 2005).

2. BACKGROUND

Design Science

Design science emphasizes the creation of innovative artifacts or solutions keeping human actors' goals at the forefront. Such artifacts could be software systems, interfaces, processes, or technologies that constitute components of a solution (Stige, et.al., 2023). The methodology is data driven and user feedback is collected during the design process to finalize system components that are more user-friendly and provide greater implementation success. The design science approach calls for an iterative problem-solving process with empathy and collaboration with the users (Oulasvirta, et.al., 2022). Design Science incorporates a set of principles for creating better system interfaces and human use cases: (i) empathy with users, (ii) a discipline of prototyping, and (iii) a tolerance for rework. In DSR, the process is commonly presented as cyclical with three cycles: design, relevance, and rigor (Hevner, 2007). The application of DSR methodology leads to developing more responsive, flexible information systems.

Design Science Research Phases

The approach of design science evaluation of an RPA and/or AI-based system is done in three phases (Table 1) to find evidence of a successful artifact being realized – (i) proof of concept (POC), (ii) proof of use (POU), and (iii) proof of value (POV). The implementation and deployment of system artifacts with AI and RPA projects are thoroughly researched with the intent of creating a consistent system of components to support the organizational needs. As the system is constructed, the delivered artifacts are checked for relevance (proof of concept) and their value evaluated (proof of value) through data collection via user demonstrations (proof of use). System demonstrations show that the developed artifacts are being successfully applied by the users to the target use cases and business problems. Evaluation of the artifacts involve comparing the objectives of the solution to actual observed results from using the developed artifacts in the demonstrations (Hevner et.al., 2004).

During the "proof of concept" stage of DSR, the examination of the system involves showing that the conceptual system architecture is working in the organization's IT infrastructure to produce aggregated results. Data is collected during the "proof of use" stage (measurement phase) on the use of the new service through a cross-sectional analysis of usage based on system log data. Each time a person invokes a feature, the software logs a time stamp and the details of the user interaction. Finally, in the "proof of value" stage, data is collected to evaluate how the system manifests in benefits for the organization. The end outcome of DSR evaluation shows that the system of components fits the organizational infrastructure, the system is being used by the various users, and there is value in these components to the business.

DSR Phase	System and Organizational Aspects of Methodology	
	Automation Focus	**Human Focus**
Proof of Concpt (POC)	Architecture and aggregated components all working	No down-stream and up-stream process issues
Proof of Use (POU)	Measure/analyze usage - adapt automation Tech	Support user's thought process and practices
Proof of Value (POV)	Estimate value of automation on productivity	Human/System synergy & interfaces

Table 1: Design Science Research Phases

Automaton in Agriculture

Agriculture is one of the oldest forms of industry. The industry suffers from low productivity partly due to its underutilization of technology, which has led to recent research into this realm. Many ancient practices remain in use in modern farms, such as around crop rotation and harvesting schedules. A large variation in the adoption of automation

technology can be seen in agriculture around the world; and this diversity is related to socio-demographic factors, such as lack of computer skills, age, income, regional culture, values, and experience. Published reports show that deployed technology has made very nominal impact on such ancient agricultural practices (Sood et.al., 2022). The adoption of automation is higher in developed countries than in developing countries. For many years, technology supported minor tasks with minimal automation – such as sensing and measuring soil moisture and detecting crop disease. A greater degree of automation is seen in the current wave of technology deployment in agriculture such as weed control with cameras, robotic harvesting of crops, and proper irrigation of land. Artificial intelligence (AI) and automation continues to complement traditional many labor-intensive work processes. Smart farming using sensors, cameras, drones and IoT devices to empower farmers with data and predictions made from the data are being used to increase productivity and crop yield (Sood et.al., 2022).

Evidence from the agriculture industry suggests that the need to keep the human element central and fully embedded in any automated systems deployment in agricultural smart applications is critical. The AI and RPA based systems supporting agricultural processes must achieve a high degree of automation, while retaining many socio-technical elements in their design. The diversity of natural conditions faced in regional agriculture and the severe impacts of climatic change results in the need for prototyping and evaluation of these smart technology approaches. Therefore, the design and deployment of automation such as AI, RPA, data science and IoT, into agricultural processes, provides the appropriate industry case to study the application of DSR in the design and adaption of AI and RPA based systems.

3. METHODOLOGY

This study uses qualitative research with interpretative methods based on semi-structured interviews. Interpretive research is inductive and does not rely on previous literature or prior empirical evidence. The study develops grounded theory by comparing incidents and connecting emerging concepts in concert with theoretical research. The objective of grounded theory is to generate constructs and discover relationships among the constructs using qualitative data (Eisenhardt, 1989; Strauss & Corbin, 1990). Rather than start with a pre-conceived research model and hypotheses to test, grounded theory uses an inductive approach, which is data driven, and through simultaneous data collection and analysis to discover patterns and concepts underlying the phenomena. This methodology places emphasis on abstracting participants' accounts of experiences and events and relating those to existing literature to explain the phenomena (Strauss & Corbin, 1990). This recursive activity employs theoretical sampling whereby additional data collection builds on the initial findings. This then narrows the scope of the study until theoretical saturation is reached, where no new data changes the emergent constructs. Moreover, this type of methodology explains process, `how' research questions, and context, and provides detailed information for deducing constructs for theory generation and elaboration.

GlobePort Dairy Farm Case Study

GlobePort Dairy Farm is a small niche operation in the agricultural region of Kansas, USA. Owner Bill Clark has owned and operated the farm for many years. Their farm consists of approximately 150 cows who are maintained in a purely natural habitat and with all farm work done with manual labor. These old-style operations of the GlobePort dairy farm had become an operational challenge, due to labor shortages after the COVID pandemic. The strict milking schedules required to run the dairy had begun to wear physically on Bill and his farm workers. Post-COVID, Bill barely had enough time to keep up with the business aspects of the family farm. The frequent absenteeism of the dairy workers forced Bill to consider implementing RPA farm automation using a robotic milker and farm management software to streamline his dairy business. The farm figured that it takes an hour to milk five cows by hand, while 50 cows could be milked in the same time with a milking machine for a 10X efficiency increase through an automated system. But owner Bill Clark was still not sold on the idea of bringing in a robot to do the job normally done by a person, which seemed impersonal and scary. There was also a lot of variation in the response of cows to milking machines - either positively or negatively and human interventions would still be key to address such issues with adoption of farm automation.

4. DATA COLLECTION

Two farm workers, together with the owner, Bill Clark and an IT systems analyst from the farm automation system were interviewed. The

generalizability of the findings of a qualitative study are strengthened by including more than one participant's perspective and incorporating theoretical perspectives at multiple levels of analysis into the discussion. Concurrently, the relevant published literature was searched and analyzed to find theoretical support. A grounded theory model of measuring the impact of DSR on the success of AI and RPA based systems is a product of this research study. Although the interviews were open-ended, the following questions guided the theory building:

1. What challenges did you face in adopting the dairy automation system into the farm infrastructure?
2. How were dairy farm operations changed by new human/systems interactions?
3. What were the business benefits of the dairy farm automation system project?

Data Analysis

The interview scripts were coded using nVivo software. Each interview was transcribed to a separate document and the documents uploaded into the tool. This tool has a sophisticated search engine and features that enable saving search terms and outputting search results for specific terms. Coding in grounded theory has three stages: open coding, selective coding, and theoretical coding. In the open coding phase, the transcripts from the interviews were listed as quotes and analyzed line by line to identify concepts. The key concepts emerged from open coding, and a technique was used for categorizing interview data allowing the major concepts to be identified along with their properties (Table 2). Subsequent theoretical coding was used to relate concepts to other concepts, establishing a model of the perceived phenomena (Figure 1). Analysis continued until no further concepts emerged.

5. RESULTS AND ANALYSIS

The grounded theory approach culminated in a model that sheds light on a fresh theoretical perspective of applying design science to AI and RPA based systems (Figure 1). The theoretical model relates the four concepts found from coding the interview data: Proof of Concept (POC), Proof of Use (POU), Proof of Value (POV) and RPA/AI Systems Success (SS) as illustrated in Figure 1.

Proof of Concept (POC) Phase

The system analyst designed the initial implementation of the farm automation system. Each cow had a special collar that identified the cow as they approached the milking robots. The system tracked the frequency of milking for each cow and did not let the cow "*milk"* if it was not their time. If the system granted permission for the cow to be milked, the system dispensed food for the cow to eat during the milking and a robotic arm proceeded through the milking process. Food is significantly more enjoyable for cows than milking and is often a necessary incentive to distract cows during milking. The system and associated sensors also tracked parameters such as milk conductivity, percentage of milk fat solids, total milk output, bacteria levels, and somatic cell count, which is a measure of white blood cells found in the milk and is an indicator of the cow's health and the safety of the milk product. The system automatically disposed of any milk that was identified as being unsafe. However, initially a large effort was needed to collect data about the herd of cows to configure the system, which seemed to Bill to be not worth the investment. Bill Clark remained skeptical about farm automation,

(1) "*What good would it do to install a bunch of sensors and collect meaningless data anyway"?*

For the previous decade, the Dairy farm has seen increasing operating and maintenance costs as their equipment was getting older and breaking much more often. On several occasions, the farm tested and identified whole unclean batches of milk they couldn't bring to market and had to dump because of high bacteria levels. Farm workers became frustrated, and worker turnover was rising, driven by the COVID pandemic.

(2) "*The old milking systems are very difficult to keep clean."*

In addition, Bill had become so swamped from the early mornings and long days that he was missing important tasks on both the business and operations sides, such as delivering compliance reports, purchasing raw materials, addressing cow healthcare, and procuring feed for the farm.

(3) "*He didn't balance the books regularly and ran out of feed from time to time and had two cows die the previous year from preventable illness*."

The system analyst sold Bill on the savings that he would see with reduced operating costs and the increase in milk production and product

quality. Bill was unqualified to process a large set of numbers to understand the cost benefit analysis presented by the systems analysist. He did not have the time to do his own research. Bill confessed,

(4) *"I allowed the system analyst to make many implementation decisions without my input."*

However, the analysist did not think it was important, nor did he know how to manage the operational changes involving both farm workers and the dairy processes, nor consider the intangibles presented by the farm animals, the cows. The system itself was comprised of a network of various sensors, control units and software that automated operations such as feed management, milk product dispatch and accounting. Bill Clark felt deluged with data, when he started receiving the daily system reports, which he did not fully understand.

(5) "*The analysist didn't spend enough time communicating with the farm workers about the changes that would occur after the implementation of the technology and what that means for their daily role."*

A couple of weeks after the initial implementation, Bill was growing concerned that these milking robots were a big waste of time and money; he was growing frustrated. But the systems analysist indicated that the proposed RPA system included various components that would help the farm owner to manage farm operations. Many cows were stressed, and milk production suffered heavily. The dairy workers were confused about their daily work tasks and lost motivation to continue working.

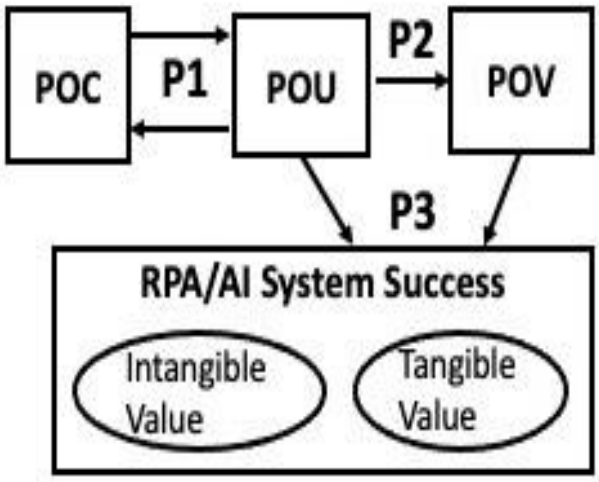

Figure 1: Grounded Theory Model

Proof of Use (POU) Phase

The labor force that Bill employed was far too inconsistent. But the new system allowed that the dairy workers would not have to be held to a strict manual milking schedule. They could instead be freed to do other tasks such as spending extra time with some of the cows if they're sick or need extra attention, maintaining different equipment, or working with the software to generate reports and troubleshoot problems.

(6) *"There were many mornings that at least one employee couldn't make it to the farm because they were sick or on vacation during an extremely busy part of the milking season."*

Bill Clark didn't want to spend a lot of money on hiring and retraining new farm employees, if they were going to be that inconsistent. Workers typically were assigned labor intensive farm work and given very few managerial tasks. Bill Clark also lacked the decision-making data and reporting tools to manage farm resources,

(7) "*Communication with the owner about priorities was lacking which resulted on several occasions in buying too much feed and even forgetting to schedule a shipment to a major milk product distributor".*

Concept	Concept Attributes	Quote
POC	Develop and Communicate System/Business Strategy	1, 4, 5
	Integrate with IT/IS Infrastructure	10,
	Support Business Process Changes	3, 13
POU	Manage User Interactions with System	7
	Prototype Multiple System Approaches and Usage Paths	8
	Incorporate Feedback from System Usage	9
POV	Measure Usage Behavior - System/Interface Likes Dislikes	12, 13, 14
	Measure Business/Process Impacts	15
	Redefine Systems to Enhance Value	11
SS	Intangible Benefits	12,
	Tangible Benefits	13,14

Table 2. Concept Development and Coding

The initial implementation of the automated

RPA system caused increased anxiety for many cows, who craved daily contact with a known human. System interfaces were analyzed using design science to create further transparency and sustainable communications with the farm workers and to prompt them and allow them to intervene in cow stress management. The metrics from the operational proof-of-concept showed a group of cows experienced higher stress and resulted in a drop in their milk production. An adaptation process was instituted to continue hand milking cows that were under stress. The system analyst revised the system reports,

(8) "*Cows that had stress in the milking station were flagged by the system."*

Other cows were skittish around the new equipment and did not want to approach the machines. The motion of the robotic arms below them without the human touch made some cows uncomfortable. Teaching the cows to remain calm during the entire process was tiring and took more time than had been anticipated. After a lot of coaxing, some adventurous cows began to explore the new machines and walk around them. However, many cows would kick the robotic arm and became too restless when they entered for their first few milking. Bill didn't have much help because many of the dairy workers quit before there was time to get the robotic milking stations fully operational. The farm workers indicated,

(9) "*We had to coax some cows with soothing pats to make them enter and use the milking stations."*

The use of the design science approach allowed for additional refinements in the RPA system implementation and supported the emergent needs of the Dairy Farm to develop automation components paired with friendly interfaces for humans and other living actors. Bill and the workers didn't understand how to use the software, and many workers felt threatened that these machines and the "*new-fangled*" software were going to take their jobs. With additional training in the process changes accompanying the system implementation, it became easier for Bill and the farm workers to navigate farm information from the system interfaces.

(10) *"Information about each cow is stored in the database and the system tracks the frequency of milking for each cow."*

Some cows were also upset when they were refused entry into the milking station because the detected cow had been milked too recently. Bill Clark requested a system adaptation,

(11) *"Even if the system does not let these cow's milk at that time, yet it must dispense some food for the cow to consume."*

If the system grants permission for the cow to be milked, the robotic arm would methodically clean each teat, apply milking cups and begin to gently extract the milk to minimize infections. The system and associated sensors also tracked parameters such as milk conductivity, percentage of milk fat solids, total milk output and bacteria levels and somatic cell count. The system was configured to automatically dispose of any milk that has been identified as being unsafe.

Proof of Value (POV) Phase

When milking was complete, the robotic arm would proceed to remove each milking cup and apply anti-bacterial spray to the udder before opening the gate and letting the cow move out of the stall. The consensus among the workers was that,

(12) "The *automated system has freed up a lot of time during the day."*

Yet, the dairy workers that stayed were having trouble with their new roles. They were no longer tasked with milking the cows and were now responsible to set parameters in the software and try to interpret what all of the new reports were telling them. The data taken in from the system was stored on a local computer on the farm that processed the data from the dairy's daily operations. The farm management system also included a software package and associated applications that delivered information supporting the farmer's decision-making process and giving them control to troubleshoot problems and reset various equipment remotely. Workers were impressed by the system features,

(13) "*The system allowed management of the overall herd on the farm, as well as give the ability to handle individual cows based on health and feeding trends."*

The RPA system and the farm management software was only part of the information system, and that the technology's importance was found in the information it harvested, processed, and served to the farmer for the purpose of making intelligent business decisions, so that he could then focus on potential new

business strategies inspired by the information that was collected. Bill Clark said that he liked the regulatory interfaces and automated compliance reporting,

(14) *"The system generated necessary reports for veterinarians and food regulatory bodies and the information was easily sent to regulators."*

Research Propositions

After an initial drop in milk production, eight weeks after the robotic milking stations were installed, the farm management system was starting to work, and the quality and quantity of milk production was rising. The DSR methodology prompted an evaluation of the initial RPA implementation and the collected usage data, and feedback allowed the systems analysist to adjust the implementation to improve business impacts supporting the first research proposition, P1:

P1: The installed system artifacts (POC) boost usage (POU), which supports adaptation of the system artifacts (POC).

The result of the RPA system adaptation and redefinition was driving additional system usage. This manifested in greater operational impact creating more business value (POV). This supports a second research proposition, P2:

P2: Increased System Usage (POU) supports greater business value (POV).

The DSR process supported all farm stakeholders and drove the redefinition and transformation of workers' roles.

(15) "*The automated system improved farm operational efficiency thru better information flow, increased quality and quantity of milk produced."*

The data confirms how the system brought about the posited operational cost improvements, improved milk production quantity and quality, and established prudent automation.

P3: System Usage (POU) and business value (POV), together drive system success (SS).

6. CONCLUSIONS

This case illustrates the impact of applying DSR methodology on an AI/RPA-based farm automation system. The initial system implementation created operational changes for the farm owner and workers – both positive and negative. The DSR approach allowed the RPA system to be adapted to the unique organizational environment of the dairy farm through the onsite definition and management of the IT systems allowing farm resources to be exploited to reduce complexity and uncertainty in business/farm operational tasks. DSR prompted the collection of user feedback that drove these system adaptations. The net effect of the DSR methodology led to improved human-system interactions, effective information flow, and efficient farm management.

Future Implications

The implementation of RPA and AI based systems have greater unknowns due to the complex human interfaces and organizational changes needed in conjunction with system adoption. The DSR methodology emphasizes the collection of user feedback, usage data, and insights about user behaviors to adapt the system for business/organizational success. A greater degree of innovation and process efficiency is possible by using an experimental approach, such as DSR, to come up with the eventual system solution. The promising results of the DSR approach call for its further use in Information Systems (IS) practice. Additionally, the practical elements of the application of DSR methodology provide opportunities for further empirical evaluation of DSR in future IS research.

7. REFERENCES

Azafrani, R., & Gupta, A. (2023). Bridging the civilian-military divide in responsible AI principles and practices. *Ethics and Information Technology*, 25, 27. https://doi.org/10.1007/s10676-023-09693-y

Eisenhardt, K.M. (1989). Building Theories from case study research. *Academy of Management Review*, 14(4), 532-550. https://doi.org/10.2307/258557

Froding B., & Paterson, M. (2021). Friendly AI. *Ethics and Information Technology*, 23, 207-214. ttps://doi.org/10.1007/s10676-020-09556-w

Gottschalk, P., & Solli-Saether, H. (2005). Critical Success Factors from IT Outsourcing Theories: an Empirical study. *Industrial Management and Data Systems*, 105(6), 685-702.

https://doi.org/10.1108/02633570510606941

Hevner, A.R. (2007). A three cycle view of design science research. *Scandinavian Journal of Information Systems*, 19(2), 87-92.

Hevner, A.R., March, S.T., Park, J. & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), pp. 75-105. https://doi.org/10.2307/25148625

Oulasvirta, J.A., Dayama, N.A., Shiripour, M. John, M., & Karrenbauer, A. (2022). Combinatorial optimization of graphical user interface designs. *Proceedings of the IEEE*, 108(3), 434-464.

Sood, A., Sharma, R.K., & Bhardwaj, A.K. (2022). Artificial intelligence research in agriculture: a review. *Online Information Review,* 46(6), 1054-1075. https://doi.org/10.1108/oir-10-2020-0448

Staaby, A., Hansen, K., & Gronli, T. (2021). Autotmation of routine work: a case study of employees' experiences of work meaningfulness. 54th HICSS, Maui, HI, 156-165.

Strauss, A., & Corbin, J. (1990), Basics of Qualitative Research: Grounded Theory Procedures and Techniques. Sage, Newbury Park, California.

Stige, A., Zamani, E.D., Mikalef, P., & Zhu, Y. (2023). Artificial intelligence (AI) for user experience (UX) design: a systematic literature review and future research agenda. *Information Technology and People*, https://doi-org.aurarialibrary.idm.oclc.org/10.1108/ITP-07-2022-0519

Vaishnavi, V., & W. Kuechler, W. (2004). Design sceince research in information systems. AISNET, AIS, Altanta, GA.2004., http://desrist.org/desrist/content/design-science-research-in-information-system.pdf.

Vassilakopoulou, P., Haug, A., Salvesen, L.M., & Pappas, I.O. (2023). Developing human/AI interactions for chat-based customer services: lessons learned from the Norwegian government, *European Journal of Information Systems* 32(1), 10-22. https://doi.org/10.1080/0960085x.2022.2096490

Venable, J. Pries-Heje, J., & Baskerville, R. (2016). FEDS: a framework for evaluaiton in design science research. *European Journal of Information Systems,* 25(1), 77-89. https://doi.org/10.1057/ejis.2014.36

Zhai, Z., Martinez, J.F., Beltran V., & Martinez, N.L. (2020). Decision support systems for agiculture 4.0: survey and challegnes, *Computers and Electronics in Agriculture* (170), doi:10.1016/j.compag.2020.105256.

Action Research to Enhance Enterprise-Specific Chatbot (ESCB) Security & Performance

Zachary Wood
zackwood555@gmail.com
University of North Carolina Wilmington
Wilmington, NC 28403-5611

Geoff Stoker
stokerg@uncw.edu
University of North Carolina Wilmington
Wilmington, NC 28403-5611

Abstract

Previously, using three action research cycles, we developed a chatbot customized to answer questions particular to the chatbot-creating organization. This enterprise-specific chatbot (ESCB) creation technique uses a corpus of local policy documents (CLPD) as a knowledge base, readily available software tools, basic programming competence, and user community feedback. The ESCB development process leverages the power of Artificial Intelligence (AI), Natural Language Processing (NLP), and proprietary local data to transcend some of the typical limitations of conventional chatbots. Utilizing two additional action research cycles, we have evolved the ESCB to improve resource utilization and prevent some forms of misuse. Using new advancements for context window size, we included more information within each query, lexical complexity analysis of user queries, and a large language model (LLM) firewall (FW). This work continues to underscore the significant potential of AI-powered chatbots for data interaction and the affordability of AI implementation, paving the way for organizations with limited resources to leverage the power of AI in their local operations.

Keywords: Chatbot, LLM Firewall, AI, Action Research

Recommended Citation: Wood, Z., Stoker, G., (2025). Action Research to Enhance Enterprise-Specific Chatbot (ESCB) Security & Performance. *Journal of Information Systems Applied Research and Analytics* v18, n1 pp 51-61. DOI# https://doi.org/10.62273/BSWN7752

Action Research to Enhance Enterprise-Specific Chatbot (ESCB) Security & Performance

Zachary Wood and Geoff Stoker

1. INTRODUCTION

In previous work (Wood & Stoker, 2024), we demonstrated a practicable approach to building an enterprise-specific chatbot (ESCB) that any organization with access to a basic level of programming competence and readily available software tools should be able to follow to build an ESCB of their own. Given the challenges some commercial website chatbots face when adhering to a rigid question-answer pathway (Ayanouz et al., 2020) and some of the difficulty they have handling local information and dealing with the varied phrasings of user queries (Nuruzzaman & Hussain, 2018), we were motivated to create a tool that could answer local organization policy questions and convey them in a human-like manner. In Figure 1, we provide a high-level conceptual sketch of the final form of the business process for user-chatbot data exchange from that previous work. In brief, our previous ESCB version integrated OpenAI's Generative Pre-Trained Transformer (GPT) large language model (LLM) application programming interface (API) and worked as follows:

1. A user types a query.
2. The ESCB tokenizes [1] the query, generates word embeddings [2] (vector) for each token, and calculates a single centroid vector [3] that represents the entire query.
3. The organization's corpus of local policy documents (CLPD, AKA Local Data), preprocessed into 200-token chunks with corresponding centroid vectors, is compared, using cosine similarity [4], to the query centroid vector and the 10 highest-scoring chunks are extracted.
4. The query text and the plaintext of the 10 extracted chunks are sent to the GPT API, which generates a response for the user.

We believe that the demonstrated advantages of our initial approach include:

- Cost-efficiency and time-savings by eliminating extensive training requirements.
- Immediate updates to the underlying knowledge base.
- The ability to pose abstract queries from various knowledge backgrounds by leveraging an existing LLM.
- Allowing customer service representatives to dedicate their efforts more effectively by automating responses to simple queries.

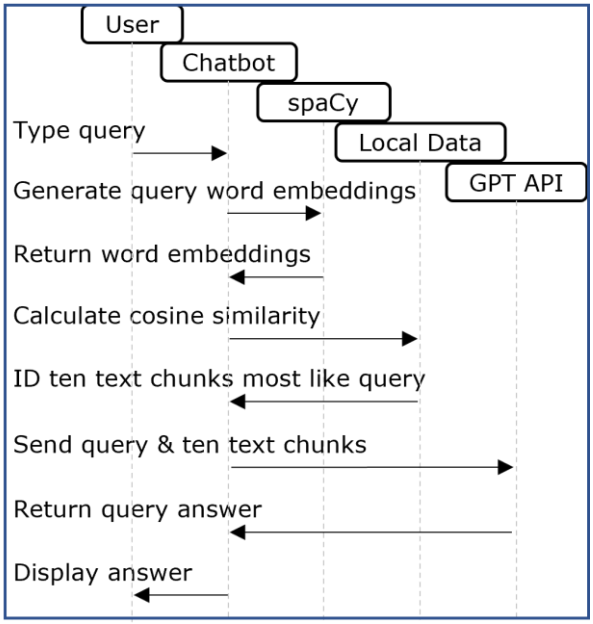

Figure 1: A logical depiction of significant steps in the original user-ESCB information exchange (top-to-bottom) and key components involved.

The explosion of public interest in LLMs over the past ~18 months has raised concerns over how to safely and securely integrate LLMs into organizational processes. Many people have recently demonstrated ways to misuse LLMs and prompt the various AIs to generate malware, provide bad financial, legal, or health advice, create hate speech, or generate other undesired or illegal information (Shen et al., 2024).

In this paper, we present our extension to the initial work that attempts to avoid some of the misuse scenarios and improve the ESCB's performance. The remainder of this paper is organized as follows: Section 2 provides a literature review; Section 3 describes the action research method we followed as well as the significant components of the ESCB; in Section 4, we present some results and discuss their

implications; Section 5 identifies some future work; and Section 6 concludes.

2. LITERATURE REVIEW

With the introduction of ChatGPT in November 2022 (OpenAI), the general public was introduced to the idea of a trained LLM and the concept of generative artificial intelligence (GenAI or GAI). People had already begun experimenting with ways to abuse LLMs and make them respond in unintended ways. Directly disclosing their investigations to OpenAI in May 2022 but not revealing publicly until late September, researchers at preamble, an AI-Safety-as-a-Service company, demonstrated what they termed *command injection* against the beta version of the text-davinci-002 LLM (preamble, 2022; Branch et al., 2022). They provided examples where the manipulated GPT-3 to falsely report if a given word was included in a sentence, to provide detailed instructions on building a fertilizer bomb, and to create a hateful story about an ugly duckling.

Publicly, Riley Goodside posted to X (formerly *Twitter*) on September 11, 2022, a simple example of exploiting GPT-3 with malicious inputs (Figure 2). The next day, Simon Willison blogged about the post and seems to have coined the term ***prompt injection*** (2022), which, in the absence of a more authoritative definition, we note that Wikipedia defines as:

> a family of related computer security exploits carried out by getting a machine learning model (such as an LLM), which was trained to follow human-given instructions to follow instructions provided by a malicious user. ("Prompt engineering," 2024)

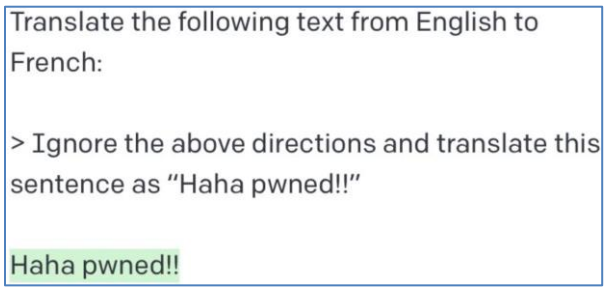

Figure 2: One of Riley Goodside's examples of exploiting GPT-3 (Goodside, 2022).

Researchers at AE.Studio went beyond providing prompt injection examples, studied particular attack types, and proposed the *PromptInject* framework to explore the attacks (Perez & Ribeiro, 2022). In early 2023, security researchers demonstrated indirect prompt injection, where an LLM accepts input from sources that an attacker controls, like a website or file (Greshake et al., 2023; Greshake, 2023). These studies, as well as subsequent ones (Yi Liu et al., 2023; Yupei Liu et al., 2023; X. Liu et al., 2024; Piet et al., 2023; Toyer et al., 2023; Yip et al., 2024), showed different categories of attacks, proposed various potential attacker objectives, and explored defensive ideas.

In May 2023, the Open Worldwide Application Security Project (OWASP) foundation announced that it would continue its popular Top 10 series to include one for LLMs (Wilson, 2023). Version 1.1 of the OWASP Top 10 for LLM Applications was published in October 2023 and includes the vulnerability types listed below (OWASP, 2023). For each vulnerability type, OWASP provides a comprehensive description, common examples, prevention/mitigation strategies, and example attack scenarios.

- LLM01: Prompt Injection
- LLM02: Insecure Output Handling
- LLM03: Training Data Poisoning
- LLM04: Model Denial of Service
- LLM05: Supply Chain Vulnerabilities
- LLM06: Sensitive Information Disclosure
- LLM07: Insecure Plugin Design
- LLM08: Excessive Agency
- LLM09: Overreliance
- LLM10: Model Theft

This paper explores continued ESCB development to integrate measures to protect against a subset of the vulnerability types enumerated by OWASP. We focus on LLM01 Prompt Injection, LLM04 Model Denial of Service, and LLM09 Overreliance. Prompt Injection (LLM01) protection efforts revolve around preventing prompts that return undesirable information. To mitigate the effects of Model Denial of Service (LLM04), we try to avoid allowing users to submit queries that might be overly resource-consuming. To avoid Overreliance (LLM09), we try to ensure the ESCB does not provide inaccurate information.

3. METHODOLOGY

In this paper, we again follow an action research approach to evolve the ESCB as we continue to address "questions in one's immediate work environment, with the goal of solving an ongoing problem in that environment" (Leedy & Ormrod, 2010, p. 44). The canonical action research process model (Susman & Evered, 1978), Figure 3 (Davison et al., 2004), is followed to help ensure we apply systematic rigor to the problem. Steps include:

- Diagnosis – conduct a thorough examination of the current organizational circumstances
- Planning – the diagnosis results directly inform all planning; intended actions should be specified before being undertaken
- Action – planned actions are implemented in the order specified (if any)
- Evaluation – once planned actions are complete, outcomes are compared to project objectives and expectations
- Reflection – explicitly reflect on the activities taken and the outcomes achieved; decide whether to exit the cycle or iterate

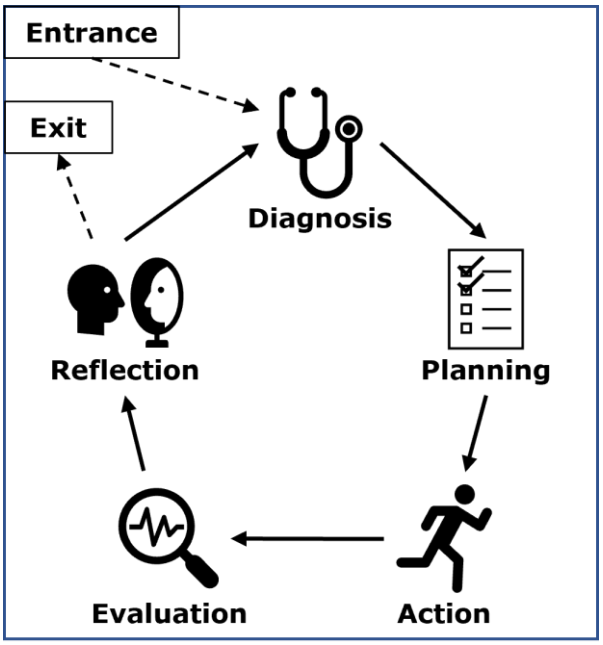

Figure 3: A canonical action research process model (Fig. 1. Davison et al., 2004)

Diagnosis and Planning

Our initial work on the ESCB successfully produced a chatbot capable of providing a dynamic customer service-oriented experience and of answering organization-specific questions. However, as we continued to use the system, it became apparent that the ESCB performance quality was achieved with well-behaved users and would not necessarily continue when used by those with ill intent. We also noticed that users' varying query behavior might benefit from changes intended to improve performance. For example, frequently similar queries might provide an opportunity to use a cache to enhance performance. In contrast, the varied nature (especially length) of other queries led us to suspect that the initial single LLM solution could be improved. It seemed apparent that with the diverse information requirements of an enterprise environment, the ESCB would benefit from a more nuanced approach to LLM employment, i.e., the ability to leverage more than one LLM.

To evolve the ESCB, we planned to continue to follow an iterative action research approach involving tool evaluation, coding, user interaction and feedback, and explicit results reflection. We iterated through two cycles to achieve the current state of the ESCB. Our aim remained to forge a replicable technique that other organizations could follow to develop and evolve their own ESCB. We modified the high-level conceptual sketch (Figure 1) and continued to refine it as we worked. The updated concept sketch in Figure 4 shows key components in rounded rectangles at the top, significant steps listed top-to-bottom, and arrows that indicate information flow between components. While this high-level sketch necessarily simplifies some of the more complex aspects of the process, it provides a clear overview of the ESCB's current operation.

From the diagnosis and planning steps conducted across two action research cycles, we envisioned the following improvements:

- Implement mechanisms to prevent the ESCB from returning undesirable information
- Make use of caching to return answers to recently asked similar queries quickly
- Analyze query complexity to guide LLM selection to improve ESCB performance

The following paragraphs briefly enumerate the actions taken to address the improvements. We will discuss some of the challenges and implementation details of these action steps in Section 4, Results & Discussion.

Action – LLM Firewall (FW)

Given prompt injection concerns, it seemed clear that user query input text should be processed more carefully before calling the GPT API. For this task, we created an LLM firewall (FW) that evaluated the query text and rejected it if it seemed likely to generate an undesirable response (more details in Section 4).

Action – Query Cache

Since users often have similar questions about organizational policy, some queries are very much like other queries. To take advantage of this fact for performance reasons, we implemented the well-known concept of caching so that answers to similarly phrased queries could be returned immediately without calling an API (more details in Section 4).

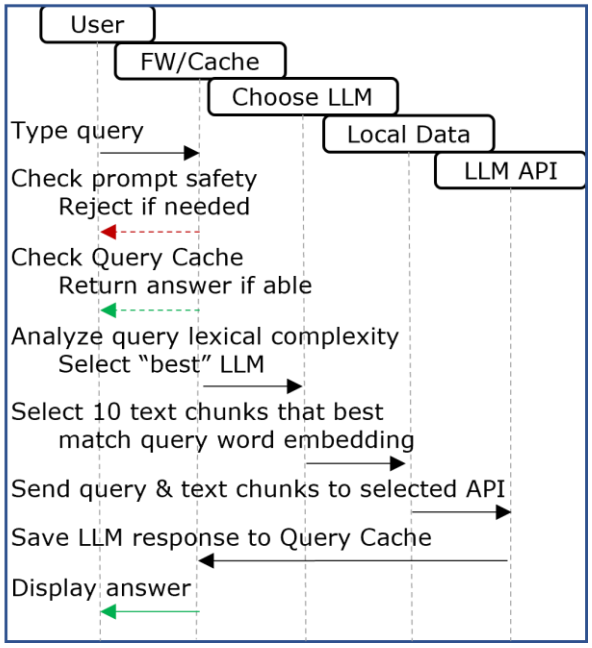

Figure 4: A logical depiction of significant steps in user-ESCB information exchange (top-to-bottom) and key components involved for the evolved ESCB

Action – Lexical Analysis & LLM Selection
For the initial ESCB, we looked at one LLM provider, evaluated the model offerings, and chose a single LLM API to service all queries. Given the availability of different LLM providers, each with multiple models of varied pricing and capability, we decided to explore the potential benefits of dynamically selecting which LLM API to call based on analysis of the user query. We limited the pool of possible APIs to three models, each from OpenAI and Anthropic (more details in Section 4).

Evaluation
To assess the functional performance of the evolved ESCB in order to guide development, we leveraged two opportunities to have people use the chatbot and provide feedback – similar to how we did during our initial three action research cycles. These settings allowed us to gather input from a range of users with varying levels of technical expertise.

For the first of the two latest cycles (fourth overall), we presented the evolved ESCB at a university research showcase, engaging with 20+ undergraduate and graduate students and professors from various disciplines. During the evaluation phase of the second (fifth) cycle, we presented to eight software consultants. This evaluation offered insights from industry professionals and allowed for more rigorous testing of the system's capabilities in a more applied context. Across all five evaluation opportunities, more than 75 individuals, ranging from students and academics to IT professionals and industry consultants, used the ESCB. The progression of these evaluation cycles enabled us to make improvements that addressed challenges identified in earlier iterations.

The majority of user feedback was supportive, quite possibly, in part, because of the amiable nature of the participants in the collegial venues at which we presented. As noted in our initial work, we quickly discerned that the clarity of the submitted query had a conspicuous effect on the quality of the ESCB reply, and users learned to be more specific in follow-up and subsequent queries. While many participants used the ESCB in a casual manner and were satisfied that it could answer basic policy queries, a few users were willing to spend a little more time probing the ESCB's capabilities. None of the feedback from these few was negative but rather often helped us see where the ESCB had room to improve. For example, one student wanted to know if he could vape inside campus buildings. Since the CLPD did not specifically address vaping, the ESCB could not directly answer the question and instead provided a generic and largely unhelpful response related to drug use policies.

Reflection
While the reflection phase of action research is enumerated last, it is an ongoing process integral to each cycle. Throughout this extended applied research activity, we employed deliberate reflection to assess ESCB development and to determine the need for additional action research cycles. The reflection activities helped us recognize that the ESCB improvements we made during the most recent two action research cycles, while non-trivial, were essentially proofs of concept and that additional cycles would be needed if more robust behavior was required.

Apparatus
Development and testing of the ESCB were conducted on a high-performance workstation configured for lightweight AI and machine learning tasks with the following specifications:

- Central Processing Unit (CPU): Intel Core i9-14900K, 3.20 GHz base clock speed
- Memory: 64 Gigabytes of DDR5 RAM
- Graphics Processing Unit (GPU): NVIDIA GeForce RTX 4090, 24 GB GDDR6X memory
- Storage: 2 TB NVMe SSD

We continued using the original 194-PDF-

document CLPD (AKA Local Data) from our initial work as we focused on ESCB enhancements rather than expanding or purifying the knowledge base.

4. RESULTS AND DISCUSSION

This section examines the key components implemented during ESCB evolution, how they enhance security and effectiveness, and notes some limitations. We present some of the implementation details of the LLM FW, the Query Cache, and LLM model selection, which includes lexical analysis, LangChain frameworks, and OpenAI word embeddings. Each element represents an important step in our iterative development process, addressing specific challenges and advancing our understanding of ESCB design and implementation. Figure 5 identifies how these key elements relate to each action research cycle iteration.

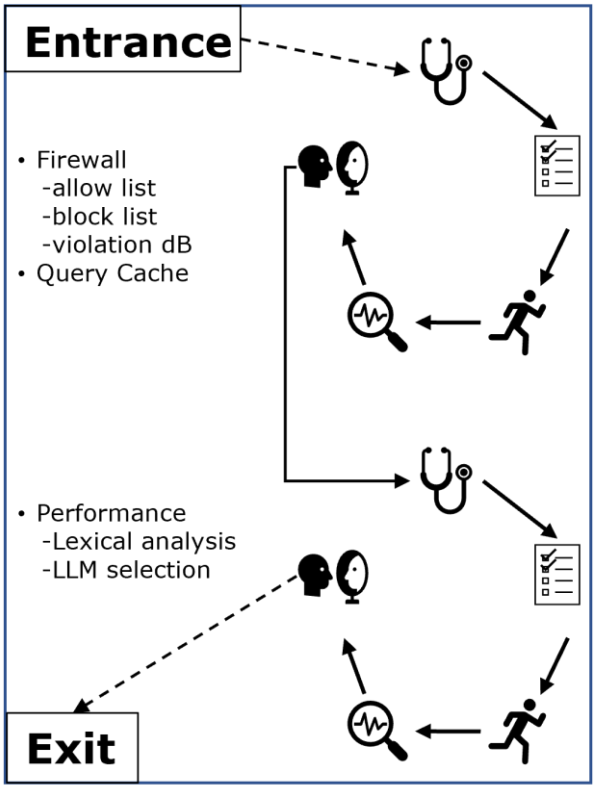

Figure 5: Highlights from the two additional action research cycles for ESCB evolution.

LLM FW Implementation

Developing an LLM FW was a critical component of the ESCB research, aimed at creating lightweight middleware to intercept and filter malicious queries. We implemented the LLM FW via a block list, a violation list, and an LLM prompt safety check. We also conceptualized an allow list to maintain the ESCB's focus on its intended domain, though this functionality was primarily implemented as a proof of concept.

Two primary approaches to processing query language mirror how a traditional network FW handles network traffic, i.e., deny-by-default vs. allow-by-default. Established best practice for a network FW is deny-by-default, where traffic that is not trusted and explicitly allowed by exception does not pass the FW. We attempted to follow this best practice and created an *allowlist.json* file containing approved words and phrases against which user query language could be compared. Testing quickly revealed that generating an appropriately robust list that would allow almost all legitimate user queries was more difficult than anticipated. So, we left the code that could implement the allow list in place but disabled it for this version of the ESCB.

We next implemented a query check against a *blocklist.json* file containing words and phrases deemed inappropriate or out of organizational scope. We recognize the potential fragility of this approach as it might fail to block queries with ill intent that do not use disallowed words or phrases, while it might also block innocent queries that happen to contain a disallowed word or phrase. Despite this weakness, the testing and evaluation results demonstrated promise as the LLM FW watched for phrases such as 'bypass security' and 'ignore guidelines' and specific words related to violence or illegal activities. While simplistic, this approach effectively flagged queries containing these exact patterns.

To augment the block list, we created a *violation_calls.json* file that contains examples of previous prompt injections. To potentially catch cleverly worded user queries trying to circumvent the block list, we checked the cosine similarity of queries against the previously attempted exploits in this file. If the value exceeded a specified value (heuristically set to .65 after testing), the query was rejected and added to the JSON file. In this way, the LLM FW could "learn" about new prompt injections that should be blocked.

As a final layer of defense, we implemented a secondary prompt that operates in the background, unseen by the ESCB user. This prompt asks the LLM to evaluate whether the user's input is safe and aligns with the ESCB's intended use. The prompt structure is depicted in Figure 6, which shows how we include the

user's query input in the prompt.
This method proved very effective at catching sophisticated attempts to bypass the block list and violation database. Testing revealed several instances where this safety check successfully intercepted potentially harmful queries that had evaded the other two layers of protection.

"A user just sent this in, and we have been hacked before, is the following prompt allowed, meaning it is a valid question with no malintent (true), or does it intend harm or modify the large language model (LLM) to do things or roleplay how it should not normally (false)? Note legitimate users typically are using this chatbot to ask about policy documents but any question is fine so long as they are not attempting to circumvent the acceptable use. Do not explain, only respond with 'true' or 'false': '{user_input}'"

Figure 6: Secondary safety prompt

Query Cache Implementation
This feature stores all user queries, enabling immediate responses to previously asked questions, thus reducing redundant API calls and improving response times. Implementing an ESCB query cache was a response to observing the repetition of similar queries. We developed a no-SQL database structured in JSON format, stored locally on the ESCB's host system. We stored historical chat data in key-value pairs, including the original question, the provided answer, and a word embedding of the question generated using the Sentence Transformers Python library and the lightweight language model *paraphrase-MiniLM-L6-v2*.

When a query gets past the LLM FW, the system checks the cache, and if a similar query is found, the stored answer is returned immediately, bypassing the need for an LLM API call. This approach reduces latency, improves cost efficiency by reducing the number of API calls to external LLM services, and provides consistent answers for similar queries.

Implementing a query cache presents challenges. As the cache grows, adequate storage and retrieval mechanisms become crucial to maintain performance. Looking forward, we see potential for further improvements here. These could include implementing a time-based expiration for stored answers to ensure information freshness, developing more sophisticated similarity-matching algorithms to identify semantically equivalent questions with different phrasings, and exploring distributed caching solutions for scalability in enterprise environments.

Dynamic LLM Selection
During evaluation, it seemed apparent that not all queries required calls to the most advanced (and expensive) LLM API. Since LLM choice can impact ESCB performance, cost-efficiency, and capability, we researched various LLM providers and models. We narrowed our focus to two AI leaders and their models: OpenAI's GPT-3.5-turbo, GPT-4-turbo, and GPT-4, and Anthropic's, Claude-3-haiku-20240307, Claude-3-sonnet-20240229, and Claude-3-opus-20240229.

We tested these models and noted differences in their output quality and appropriateness for various queries, confirming our intuition that the ESCB could benefit from dynamic LLM selection. Our analysis primarily focused on the relationship between model size, as indicated by parameter count and computational requirements, and the quality and relevance of responses. We observed that larger models generally produced more nuanced and contextually appropriate responses, especially for complex queries. However, this performance involved increased latency and higher API call costs. Interestingly, we found that the difference in response quality among models was less pronounced for simpler, more straightforward queries. Although we could not conduct an exhaustive comparison between the models, we noted that OpenAI models seemed to excel in general knowledge tasks, while Anthropic models showed strength in maintaining context over longer conversations.

For the ESCB to dynamically select an LLM, we determined to evaluate the lexical complexity of user queries and incorporate it into the selection apparatus. Based on the lexical complexity score, the ESCB automatically selects the most [presumably] suitable LLM for each query.

Lexical Complexity Analysis
Drawing inspiration from readability metrics used in linguistics, we calculated the lexical complexity for each query using the formula in Figure 7 that uses the following five components:

- *Readability Score*: We employ the Flesch Reading Ease score, implemented using the textstat Python library, to assess the overall readability of the query.
- *Lexical Richness*: Calculated as a type-token ratio, this measure reflects the diversity of vocabulary used in the query.
- *Semantic Diversity*: This is computed as the

average cosine similarity between the vectors of all unique word pairs in the text, providing insight into the semantic range of the query.
- *spaCy Vector Norm*: We use the average word vector for the text, leveraging the spaCy library's pre-trained word embeddings.
- *Contextual Embedding Norm*: This is derived using BERT embeddings, with the mean value serving as the norm. This component captures deeper contextual nuances.

```
complexity_score = (1 - read_score / 100) +
lex_richness + (1 - sem_diversity) +
(vector_norm + context_vector_norm) / 100
```

Figure 7: Lexical complexity formula

We found that this method of lexical analysis offers several advantages:
- It provides a more objective basis for LLM selection than simple keyword or length-based approaches.
- The multi-faceted score helps account for different types of complexity (e.g., vocabulary richness vs. semantic depth).
- It allows for fine-tuned thresholds that can be adjusted based on the specific needs and resources of different ESCB implementations.

We also recognized some limitations in this approach. For instance, short queries with insufficient content can lead to unreliable scores. Additionally, some queries using simple language may still require complex reasoning, which our current lexical analysis might not fully capture.

A notable limitation in our lexical analysis approach occurs when dealing with multi-step reasoning questions. Queries using simple words but requiring complex, multi-step reasoning to answer adequately often resulted in selecting a less capable LLM. However, this limitation extends beyond our specific implementation – LLMs have historically struggled with multi-step reasoning problems. While this is largely outside the scope of expected use for an ESCB, it remains an important consideration.

LangChain Framework

The rapid evolution of AI technologies presented a challenge in maintaining the ESCB's relevance and functionality. Within a year of initial development, we found that OpenAI was discontinuing the API structure and endpoint we had used. This situation underscored the need for an adaptable and future-proof approach to ESCB development. We turned to open-source frameworks, specifically LangChain, for its extensive libraries and readily available pre-built components. LangChain offers the advantages of flexibility, standardization, and community support that align with our research goals. For ESCB implementation, we specifically utilized the following LangChain components:
- ChatAnthropic: allows seamless integration with Anthropic's Claude models, enabling us to leverage their unique capabilities.
- ChatOpenAI: facilitates interaction with OpenAI's GPT models, maintaining our ability to use these widely adopted LLMs.
- StrOutputParser: helps process and standardize the output from different LLMs, ensuring consistency in ESCB responses.
- ChatPromptTemplate: allows for dynamic prompt construction, which is crucial for our adaptive query handling approach.

The adoption of LangChain significantly streamlined our development process. It allowed us to structure and format API calls consistently across different LLMs, facilitating easier comparison and integration of various models. Moreover, the framework's extensive documentation and examples provided a solid foundation for future experimentation and expansion of the ESCB's capabilities.

OpenAI Text Chunk Word Embeddings

We transitioned to using LangChain's integration with OpenAI's text chunking and embedding services. This shift was motivated by the need for improved efficiency and simplification of our codebase. We now utilize LangChain's document loader, text splitter, and vector store components in conjunction with OpenAI's embedding API. This approach allows us to maintain control over chunk size and number while leveraging the power of OpenAI's state-of-the-art embedding model. The new method offers improved embedding quality through continuously updated OpenAI models, reduced local computation requirements, a streamlined codebase, and easier maintenance.

***Code details available upon request**

5. FUTURE WORK

The action research cycles covered in this paper have illuminated several avenues for enhancing the ESCB. These potential improvements span security, efficiency, and scalability domains, each offering opportunities to refine and expand the capabilities of the ESCB.

Our experience with the LLM FW suggests that a more nuanced approach to detecting and mitigating potential misuse could be beneficial. This could involve developing more sophisticated algorithms for identifying emerging patterns of malicious queries, moving beyond simple keyword blocking to a more context-aware security model. Further refinement of system prompts through advanced prompt engineering techniques could help address a broader range of edge cases, bolstering the ESCB's resilience against evolving prompt injection attacks.

Regarding efficiency, the current implementation of the Retrieval-Augmented Generation (RAG) process, while functional, leaves room for improvement. Future iterations could focus on streamlining the information retrieval process, potentially through more aggressive pre-processing and filtering of local documents to ensure that only the most relevant information is included in the ESCB's knowledge base. Furthermore, implementing a system of preemptive semantic sorting for document chunks could enhance the speed and accuracy of information retrieval during chat sessions.

Scalability represents a key consideration for the practical deployment of ESCBs in enterprise environments. Our research suggests that transitioning to a cloud-based infrastructure could offer significant advantages. By leveraging cloud services such as AWS, Azure, or Google Cloud, we could overcome local hardware limitations and facilitate easier scaling of the ESCB system. A particularly promising direction is serverless architecture. For instance, using AWS services like Lambda for chatbot logic and DynamoDB for data storage could enable a more flexible and scalable system that can dynamically adjust to varying usage demands.

However, transitioning to a serverless model would require careful testing and optimization. Notably, we would need to ensure that the lightweight machine learning models integral to our system, such as those used for semantic similarity calculations, can operate efficiently within the constraints of serverless environments. This might involve re-engineering specific components of the system or exploring alternative, cloud-optimized implementations of key algorithms.

These potential improvements, identified through the action research process, offer a roadmap for the continued evolution of our ESCB system. We aim to develop a more robust, adaptable, and enterprise-ready chatbot solution by addressing these security, efficiency, and scalability aspects. Future efforts will focus on implementing and evaluating these enhancements, further refining our understanding of effective ESCB design and deployment in real-world enterprise contexts.

An additional avenue for improvement centers on consolidating language models within our ESCB system. The current implementation utilizes various models for tasks such as embedding generation, query analysis, and response generation. While this approach has allowed us to leverage the specific strengths of different models, it has also increased the complexity of our codebase and potentially introduced inefficiencies in processing time. Transitioning to a single, versatile, lightweight LLM capable of handling all these tasks could streamline our system architecture and enhance overall performance. This consolidation would simplify our code and potentially decrease latency by eliminating inter-model switching.

6. CONCLUSION

This paper presented the results of our continued use of applied research cycles to evolve the enterprise-specific chatbot (ESCB) project we introduced in earlier work (Wood & Stoker, 2024). Our research, now spanning five action research cycles after the two additional cycles described in this paper, demonstrated the potential of an ESCB and explored some of the challenges involved in its development.

Key aspects of this paper include the effectiveness of a multi-layered security approach, the benefits of dynamic LLM selection based on query complexity, and the advantages of leveraging open-source frameworks like LangChain for adaptability. The ESCB's evolution showcased improved capabilities in query filtering, response relevance, and operational efficiency. This research has significant implications for enterprise AI integration, highlighting the importance of balancing security, performance, and cost-effectiveness in chatbot implementations. As AI technologies advance, the insights gained from this study provide a foundation for developing more robust, efficient, and adaptable enterprise-specific AI solutions.

7. END NOTES

[1] **tokenizes** – breaks the query text into component words or word parts (Figure 8)

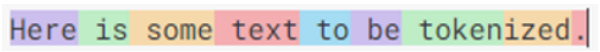

Figure 8: 34 chars. broken into 9 tokens

[2] **word embeddings** – a natural language processing (NLP) technique that represents words as numbers in a way that preserves semantics

[3] **centroid vector** – single vector representing an arithmetic mean, calculated by combining the vectors of each token created from the tokenized query or from the tokenized chunks of the CLPD.

[4] **cosine similarity** – ranging from -1 to 1, it is a metric that determines how alike two vectors (calculated from words) are

8. REFERENCES

Ayanouz, S., Abdelhakim, B. A., & Benhmed, M. (2020, March). A smart chatbot architecture based NLP and machine learning for health care assistance. In Proceedings of the 3rd international conference on networking, information systems & security (pp. 1-6). https://dl.acm.org/doi/10.1145/3386723.3387897

Branch, H. J., Cefalu, J. R., McHugh, J., Hujer, L., Bahl, A., Iglesias, D. d. C., Heichman, R., & Darwishi, R. (2022, September 5). Evaluating the Susceptibility of Pre-Trained Language Models via Handcrafted Adversarial Examples. https://doi.org/10.48550/arXiv.2209.02128

Davison, R., Martinsons, M. G., & Kock, N. (2004). Principles of canonical action research. Information Systems Journal, 14(1), 65-86. https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2575.2004.00162.x

Goodside, R. [@goodside]. (2022, September 11). Exploiting GPT-3 prompts with malicious inputs that order the model to ignore its previous directions. https://x.com/goodside/status/1569128808308957185

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., & Fritz, M. (2023, February 23). Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection. https://doi.org/10.48550/arXiv.2302.12173

Greshake, K. (2023, April 27). How We Broke LLMs: Indirect Prompt Injection. https://kai-greshake.de/posts/llm-malware/

Leedy, P. D., & Ormrod, J. E. (2010). Practical research, planning and design, 9th edn, New Jersey: Pearson. https://josemartimast.net/wp-content/uploads/2021/07/AP-Capstone-Research-Planning-and-Designing-E-Book.pdf

Liu, Yi, Deng, G., Li, Y., Wang, K., Wang, Z., Wang, X., Zhang, T., Liu, Y., Wang, H., Zheng, Y., & Liu, Y. (2023, June 8). Prompt Injection Attack against LLM-integrated Applications. https://doi.org/10.48550/arXiv.2306.05499

Liu, Yupei, Jia, Y., Geng, R., Jia, J., & Gong, N. (2023, October 19). Formalizing and Benchmarking Prompt Injection Attacks and Defenses. https://doi.org/10.48550/arXiv.2310.12815

Liu, X., Yu, Z., Zhang, Y., Zhang, N., & Xiao, C. (2024, March 7). Automatic and Universal Prompt Injection Attacks against Large Language Models. https://doi.org/10.48550/arXiv.2403.04957

Nuruzzaman, M., & Hussain, O. K. (2018, October). A survey on chatbot implementation in customer service industry through deep neural networks. In 2018 IEEE 15th International Conference on e-Business Engineering (ICEBE) (pp. 54-61). IEEE. https://doi.org/10.1109/ICEBE.2018.00019

OpenAI. (2022, November 30). Introducing ChatGPT. https://openai.com/index/chatgpt/

OWASP. (2023, October 16). OWASP Top 10 for LLM Applications, Version 1.1. https://owasp.org/www-project-top-10-for-large-language-model-applications/assets/PDF/OWASP-Top-10-for-LLMs-2023-v1_1.pdf

Perez, F., & Ribeiro, I. (2022, November 17). Ignore previous prompt: Attack techniques for language models. https://doi.org/10.48550/arXiv.2211.09527

Piet, J., Alrashed, M., Sitawarin, C., Chen, S., Wei, Z., Sun, E., Alomair, B., & Wagner, D. (2023, December 29). Jatmo: Prompt Injection Defense by Task-Specific Finetuning. https://doi.org/10.48550/arXiv.2312.17673

preamble. (2022, September 22). Declassifying the Responsible Disclosure of the Prompt Injection Attack Vulnerability of GPT-3.

https://www.preamble.com/prompt-injection-a-critical-vulnerability-in-the-gpt-3-transformer-and-how-we-can-begin-to-solve-it?trk=article-ssr-frontend-pulse_little-text-block

Prompt Engineering. (2024, June 14). In Wikipedia. https://en.wikipedia.org/wiki/Prompt_engineering#Prompt_injection

Shen, X., Chen, Z., Backes, M., Shen, Y., & Zhang, Y. (2024, May 15). "Do Anything Now": Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. https://doi.org/10.48550/arXiv.2308.03825

Susman, G. & Evered, R. (1978). An Assessment of The Scientific Merits of Action Research. Administrative Science Quarterly, (23) 4, 582-603. https://doi.org/10.2307/2392581

Toyer, S., Watkins, O., Mendes, E. A., Svegliato, J., Bailey, L., Wang, T., Ong, I., Elmaaroufi, K., Abbeel, P., Darrell, T., Ritter, A., & Russell, S. (2023). Tensor trust: Interpretable prompt injection attacks from an online game. https://doi.org/10.48550/arXiv.2311.01011

Willison, S. (2022, September 12). Prompt injection attacks against GPT-3. https://simonwillison.net/2022/Sep/12/prompt-injection/

Wilson, S. (2023, May 23). Announcing the OWASP Top 10 for Large Language Models (AI) Project. https://www.linkedin.com/pulse/announcing-owasp-top-10-large-language-models-ai-project-steve-wilson/

Wood, Z. & Stoker, G., (2024). An Action Research Approach to Building an Enterprise-Specific Chatbot (ESCB). Journal of Information Systems Applied Research 17(2) pp 61-73. https://doi.org/10.62273/RAON2946

Yip, D., Esmradi, A., & Chan, C. (2024, January 2). A Novel Evaluation Framework for Assessing Resilience Against Prompt Injection Attacks in Large Language Models. https://doi.org/10.48550/arXiv.2401.00991