Classification of Terms for Conversational Agents

What the bot?

The growing popularity of conversational agents, such as Alexa, offers new opportunities in both research and design (Følstad & Brandtzæg, 2017). The language used to describe this new context is currently large, varied and inconsistent. Although a conversational agent is a bot, bots can be anything from the basic rule-based web crawling spider to the sophisticated AI of Alexa. The voice and AI parts are new but the bot part is not (Radziwill & Benton, 2017). So, Alexa could be considered a bot. Albeit a much fancier version than the web crawler or emailing spammer, but a bot just the same. This fact can make conducting research, and a job search, in this field challenging. It would be beneficial to gain an understanding of what language is being used to refer to what and by whom both in a colloquial sense and academically.

Research Questions.

What terms are associated specifically with voice-enabled interfaces versus conversation as a platform in general? And which ones are more commonly used by UI/UX designers versus academics?

Developing a definition.

A bot is any autonomous program that performs a repetitive task in place of a human (Knecht, 2016). A chatbot is a specific type of bot designed to engage with a user, or sometimes another bot, to provide information through either audio or text (Fichter & Wisniewski, 2017). The term “conversational agent” is synonymous with chatbot, both were developed to communicate using natural language such as English (Io & Lee, 2017).

There are basically two types of chatbots you will encounter; rules-based and those that use machine learning. A rules-based bot far more limited and provides a response based on a pre-written script

that may retrieve some information from a database. They are commonly used for customer service or simple questions like “When does the library open on Saturday?” Although a device such as Alexa utilizes artificial intelligence, most of its responses are rules-based (Fichter & Wisniewski, 2017). In contrast, a chatbot like Cleverbot generates responses based on information gleaned from past conversations (“Cleverbot,” n.d.).

The focus of my research is the voice-enabled chatbots that utilize AI. Also called intelligent agents, among other things, and include Alexa, Google Assistant, Sir, and Cortana. They combine voice-recognition, NLP processing through a cloud and a speaker. Infused with personalities, they can generate an emotional attachment with users (Portela & Granell-Canut, 2017). Being a VUI (voice user interface) they are handsfree. This is beneficial to both people with situational (i.e. cooking) and permanent disabilities (i.e. visual impairment) (Saran, 2017). And of course, using natural language as an interface, they provide a very low entry level for users.

In most cases, both text-based and devices with VUIs should be considered since they both benefit from the same backend. Amazon Lex, which can be implemented in Facebook Messenger, is driven by the same AI as Alexa on your Echo device (“Amazon Lex – Build Conversation Bots,” n.d.).

Finding a common terminology while researching chatbots, both text-based and those that use a voice user-interface, can be a struggle. As the market grows, companies look to create more descriptive and unique terminology for their products. There are new terms for designers that attempt to make a connection between current phrases within a conversational context, such as VUX (voice user experience) and Zero UIs. All mean much the same but sometimes have subtle differences.

Mapping bot terminology.

In order to determine prevalence and relationship between terms, I decided to perform a text analysis. I would need sources to build a corpus. With so many terms it was important to choose a starting point that would connect all of them. Due to the fact, there is so much variation in terminology around these devices, it was important to find a term in which to anchor my research. I chose “Alexa” due to the fact it is a proper and unique term that directly refers to the type of device that is the focus of my research (Kinsella, 2019). Amazon Alexa also has the largest market share, is well-known and is heavily invested in providing tools for designers and developers (“Alexa Skills Kit Official Site: Build

Skills for Voice," n.d.). This would make it an ideal device for a designer to become familiar with and anyone looking to conduct research.

I then chose both an academic source, Google Scholar, and a practical one, Medium, that would provide the content to build my corpus. Google Scholar is focused on scholarly material and does not include sources from news and magazine articles, reviews or editorials ("About Google Scholar," n.d.). Medium is a self-publishing platform popular with design professionals that is focused on practical material ("Welcome to Medium, where words matter.," n.d.). Content from Medium would be a good juxtaposition against the academic research reflected in the Google Scholar results.

The first step was researching Alexa and collecting terms used while describing the agent. This research was carried out in a more traditional manner since it took some judgment to decide if a term was being used to refer to the device. I looked for research, development and practical usage articles that focused on Alexa and similar agents. I settled on a list of ten terms from my research. This list of terms was used to create search strings to collect the documents I needed. Then using those terms for search, I added the top results to the corpus.

Table 1. Terms for search strings.

Smart Speaker (Kinsella, 2019)	Voicebot (Kinsella, 2019)
Conversational Agent (Luger & Sellen, 2016)	Personal Digital Assistant ("Microsoft Cortana," n.d.)
Chatbot (Lasky, 2019)	Intelligent Personal Assistant (Lopatovska & Williams, 2018)
Headless User Interface (Saran, 2017)	VUI (Voice User Interface) (H, 2016)
Zero UI (Brownlee & Brownlee, 2015)	Virtual Personal Assistant (Këpuska & Bohouta, 2018)

I created a bot that would pull up results based on the search term, then scrape the content to build a CSV file. The dataset included a unique id, article title, URL and then abstract for scholarly articles or body text for articles. The combined abstracts and body text created the corpus I used for my text analysis.

Topic modeling is a type of text analysis specifically useful when not specifically looking for any result but instead want to see what subject areas will arise when processed. I felt this would provide some insights into how the terms were being utilized.

Results

My first attempt at analysis used a single corpus of all the top results of the search terms. This did not provide clear topics even after several rounds of adding stop words. Frustrated, I took another approach where I built a separate corpus for each term and compared the results from both sources. For some terms, this gave some insight into the differences between academic documents and those on medium.

Figure 1. Three topics and terms from Topic Modeling performed on the corpus built from Google Scholar using the search term “VUI”.

Figure 2. Three topics and terms from the from the “VUI” corpus sources from Medium.com.

The search term that produced the most relevant responses was “VUI”voice user interfaces. In almost every case, the document was speaking directly to Alexa, Google Assistant, Cortana, or Siri. I performed topic modeling on both the Medium and Scholar results to compare. The academic corpus focused more on speech input, formatting (voicexml) and learning with VUIs. In contrast, Medium was focused on the devices, new opportunities for designers and skills (which is the name of Amazon Alexa apps).

The individual terms had variations between academic and colloquial but no patterns were becoming evident. At that point, I revisited my search terms. Not only did I chose ones that were used to refer to Alexa within body copy but also those that provided the most relevant search results. Using the search results form these terms, I created a corpus for the medium articles and then one for Google Scholar. This time when I ran topic modeling I had much more coherent results.

The most obvious difference between the two sets of data was the fact that the academic source produced a “health” centered topic. This may be due to the large amount of papers written on how smart speakers can help with health related issues.

I landed on five topics. The Scholar corpus focused on design of VUI interfaces, the agent, the voice functionality, technology and health.

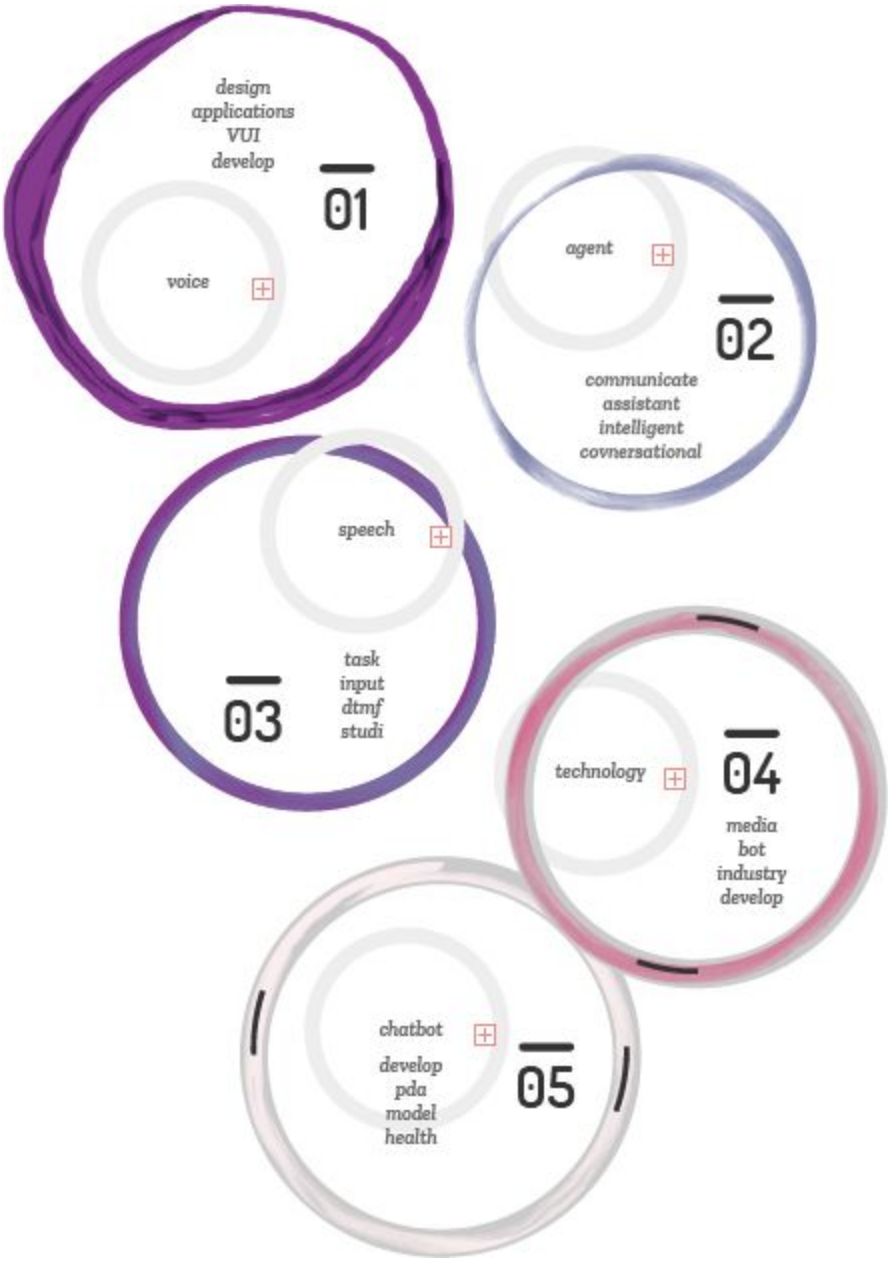


Figure 3. Five topics and terms from Topic Modeling performed on the corpus built from Google Scholar combining the top results for all search terms.

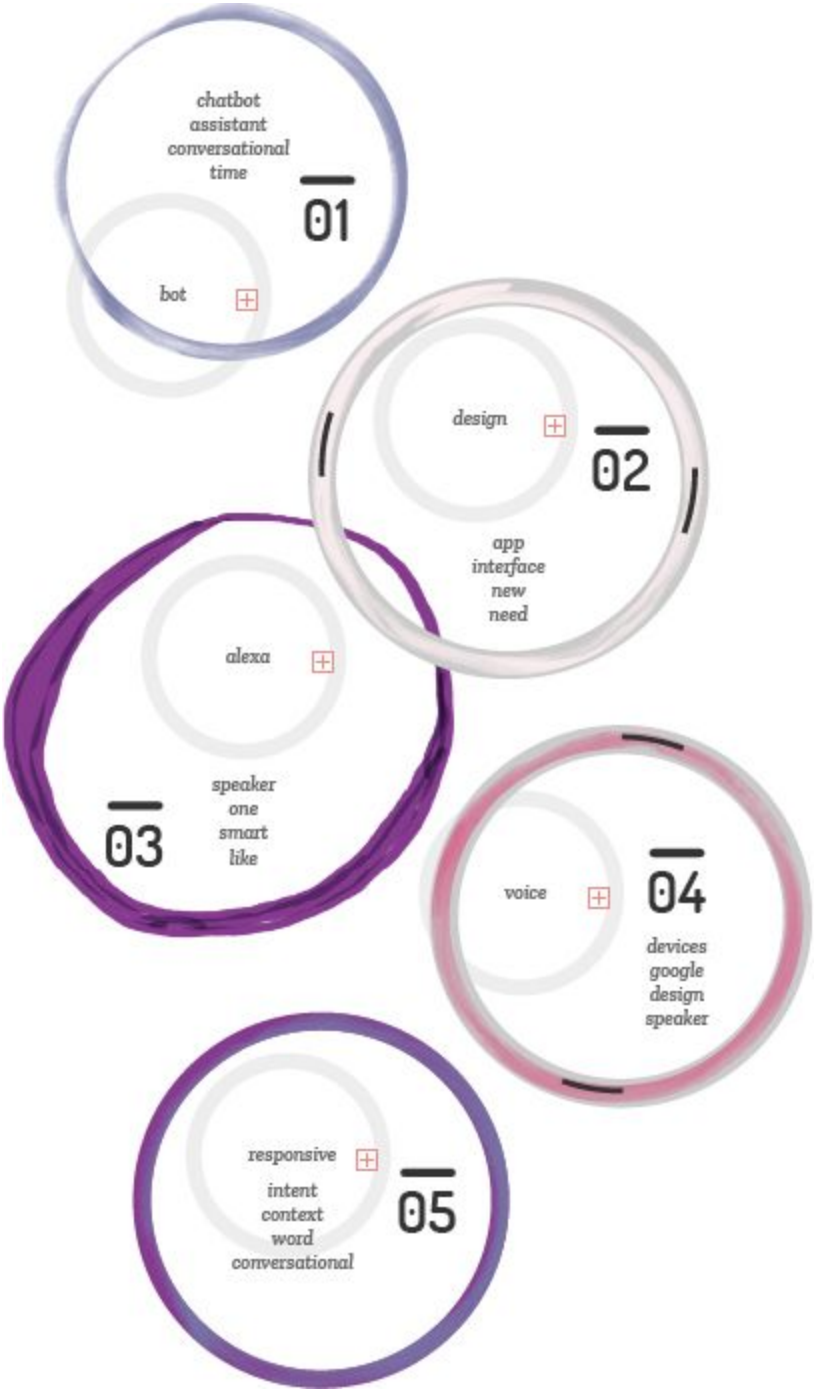


Figure 4. Five topics and terms from Topic Modeling performed on the corpus built from Medium

In contrast, the topics gleaned from the Medium articles were designed focused. The topics covered devices, conversational context, alexa smart speaker, interface design and bots. Although these differences were interesting, it was still challenging to draw conclusions from my efforts.

Further Considerations

Using the same tidy R package used for topic modeling, an analysis of word and document frequency could help us identify some themes within our corpus in relation to the search terms. Lexical density is not part of the same package, but could also help. Lexical words (content words) provide meaning and information on the subject of text. Discovering the lexical density between academic and practical writing, and, or different terms, could add another level of insight as well.

Conclusion

As the technology that powers conversational agents, machine learning and natural language processing, continues to advance (Piccolo, Mensio, & Alani, 2019), so will the need to understand and develop these interfaces. Currently, the terminology used for different categories of bots from basic rule-based web crawling spider to the sophisticated AI of Alexa is varied and can be confusing. Discovering the more prevalent terms and identifying the context in which they are used, would be a helpful contribution to the field. Building a dataset that could be used to conduct this classification would be a benefit to anyone interested in conversational agents.

Although topic modeling provided some insights, there are other forms of text analysis that may be more effective.

Bibliography

About Google Scholar. (n.d.). Retrieved December 3, 2019, from https://scholar.google.com/intl/en/scholar/about.html

Alexa Skills Kit Official Site: Build Skills for Voice. (n.d.). Retrieved December 3, 2019, from https://developer.amazon.com/en-US/alexa/alexa-skills-kit

Amazon Lex – Build Conversation Bots. (n.d.). Retrieved July 26, 2019, from Amazon Web Services, Inc. website: https://aws.amazon.com/lex/

Cleverbot. (n.d.). Retrieved November 27, 2019, from Cleverbot website: http://www.cleverbot.com/

Fichter, D., & Wisniewski, J. (2017). Chatbots Introduce Conversational User Interfaces. *Online Searcher*, *41*(1), 56–58.

Følstad, A., & Brandtzæg, P. B. (2017). Chatbots and the New World of HCI. *Interactions*, *24*(4), 38–42. https://doi.org/10.1145/3085558

Io, H. N., & Lee, C. B. (2017). Chatbots and conversational agents: A bibliometric analysis. *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 215–219. https://doi.org/10.1109/IEEM.2017.8289883

Kinsella, B. (2019, March 7). U.S. Smart Speaker Ownership Rises 40% in 2018 to 66.4 Million and Amazon Echo Maintains Market Share Lead Says New Report from Voicebot. Retrieved November 25, 2019, from Voicebot.ai website: https://voicebot.ai/2019/03/07/u-s-smart-speaker-ownership-rises-40-in-2018-to-66-4-million-and-amazon-echo-maintains-market-share-lead-says-new-report-from-voicebot/

Knecht, T. (2016, September 21). A Brief History Of Bots And How They’ve Shaped The Internet Today. Retrieved November 20, 2019, from https://www.abusix.com/blog/a-brief-history-of-bots-and-how-theyve-shaped-the-internet-to

day

Piccolo, L., Mensio, M., & Alani, H. (2019). Chasing the Chatbots: Directions for Interaction and Design Research. In s Bodrunova (Ed.), *Internet Science. INSCI 2018* (Vol. 11551, pp. 157–169). Retrieved from http://oro.open.ac.uk/57382/

Portela, M., & Granell-Canut, C. (2017). A New Friend in Our Smartphone?: Observing Interactions with Chatbots in the Search of Emotional Engagement. *Proceedings of the XVIII International Conference on Human Computer Interaction*, 48:1–48:7. https://doi.org/10.1145/3123818.3123826

Radziwill, N. M., & Benton, M. C. (2017). *Evaluating Quality of Chatbots and Intelligent Conversational Agents*. Retrieved from https://arxiv.org/abs/1704.04579v1

Saran, C. (2017). Chatbots lead the debate on public sector use of AI to streamline digital services: Chatbots have the potential to reduce the workload for contact centres. A recent Capita workshop explored the impact of artificial intelligence and other emerging technologies. *Computer Weekly*, 4–6.

Welcome to Medium, where words matter. (n.d.). Retrieved December 3, 2019, from https://medium.com/about

Brownlee, J., & Brownlee, J. (2015, July 2). What Is Zero UI? (And Why Is It Crucial To The Future Of Design?). Retrieved July 26, 2019, from Fast Company website: https://www.fastcompany.com/3048139/what-is-zero-ui-and-why-is-it-crucial-to-the-future-of-design

H, R. (2016, July 22). Usability Heuristics for Amazon Alexa and Voice User Interfaces (VUI). Retrieved September 15, 2019, from Medium website: https://medium.com/@ricardohdz/usability-heuristics-for-amazon-alexa-and-voice-user-interfaces-vui-3666dbdfc10e

Këpuska, V., & Bohouta, G. (2018). Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home). *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 99–103. https://doi.org/10.1109/CCWC.2018.8301638

Lasky, J. (2019). Chatbot. In *Salem Press Encyclopedia of Science*. Salem Press.

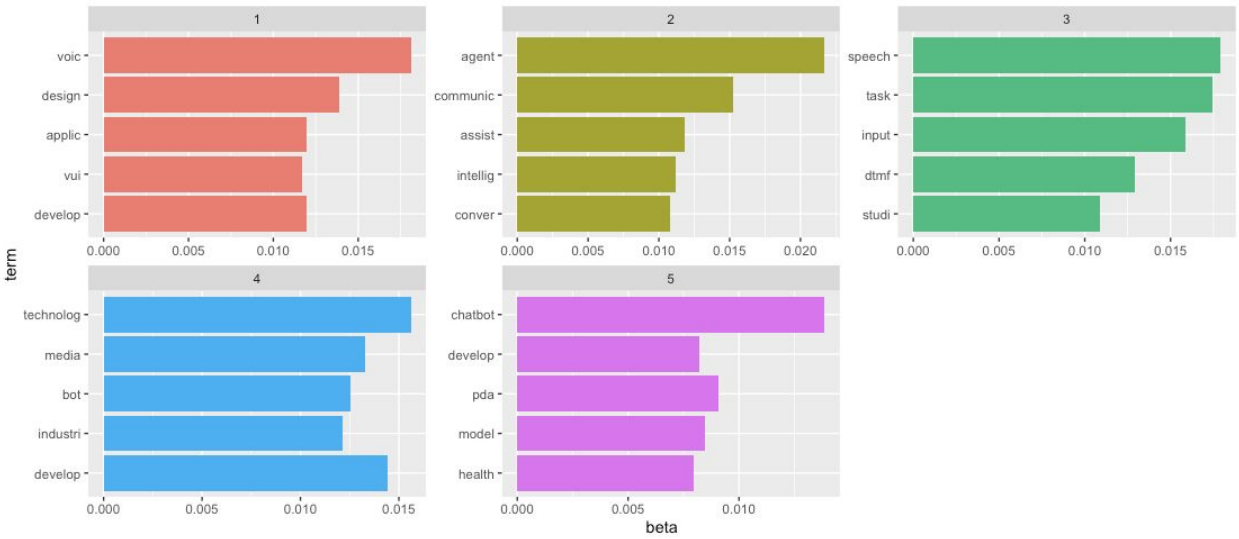
Lopatovska, I., & Williams, H. (2018). Personification of the Amazon Alexa: BFF or a Mindless Companion. *Proceedings of the 2018 Conference on Human Information Interaction&Retrieval - CHIIR '18*, 265–268. https://doi.org/10.1145/3176349.3176868

Luger, E., & Sellen, A. (2016). "Like Having a Really Bad PA": The Gulf Between User Expectation and Experience of Conversational Agents. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 5286–5297. https://doi.org/10.1145/2858036.2858288

Personal Digital Assistant—Cortana Home Assistant—Microsoft [Product]. (n.d.). Retrieved October 25, 2019, from Microsoft website: https://www.microsoft.com/en-us/cortana

Charts from R

Figure 1. Topic modeling on results from eleven search terms on Google Scholar.



z

Figure 2. Topic modeling on results from eleven search terms on Medium.

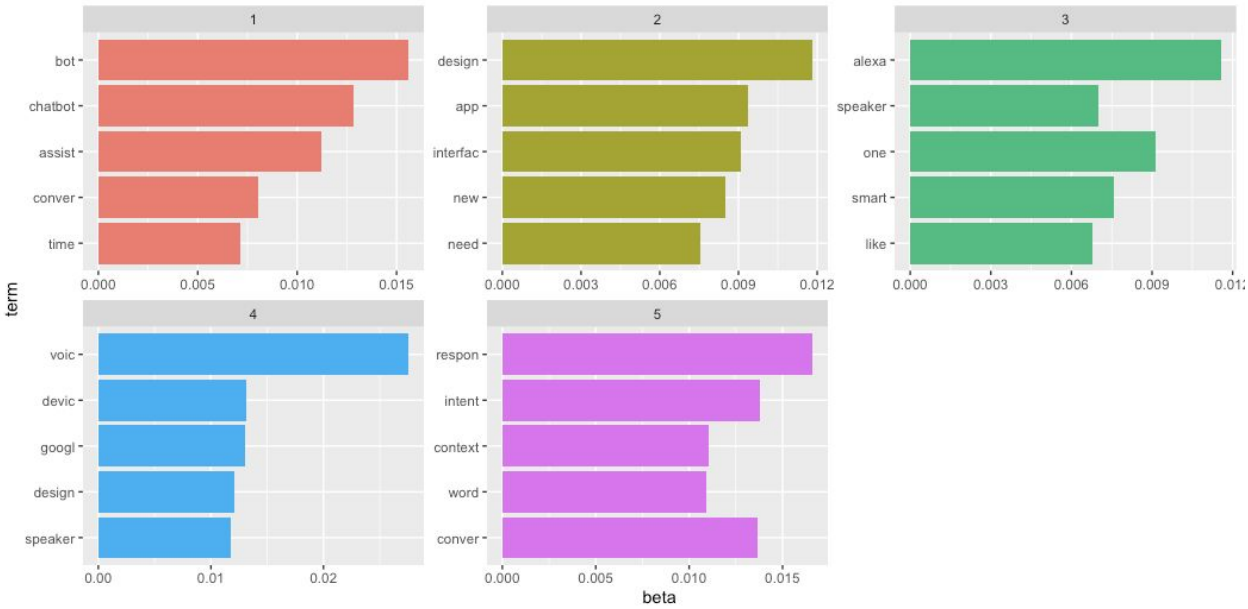


Figure 3. Topic modeling on results from “VUI” on Medium.

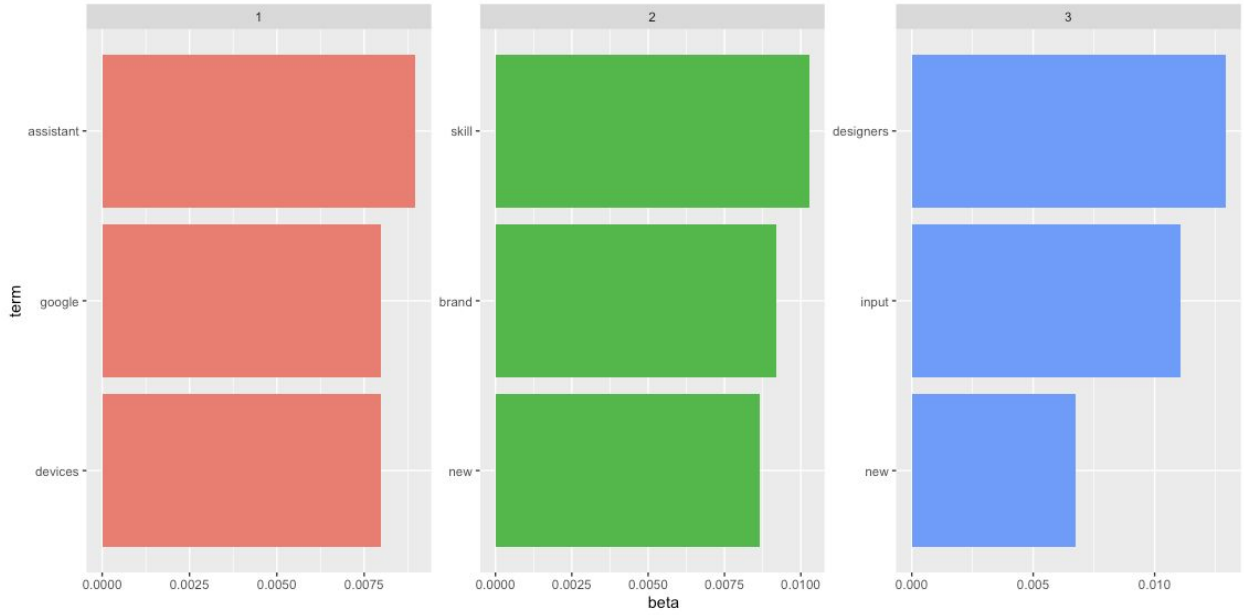
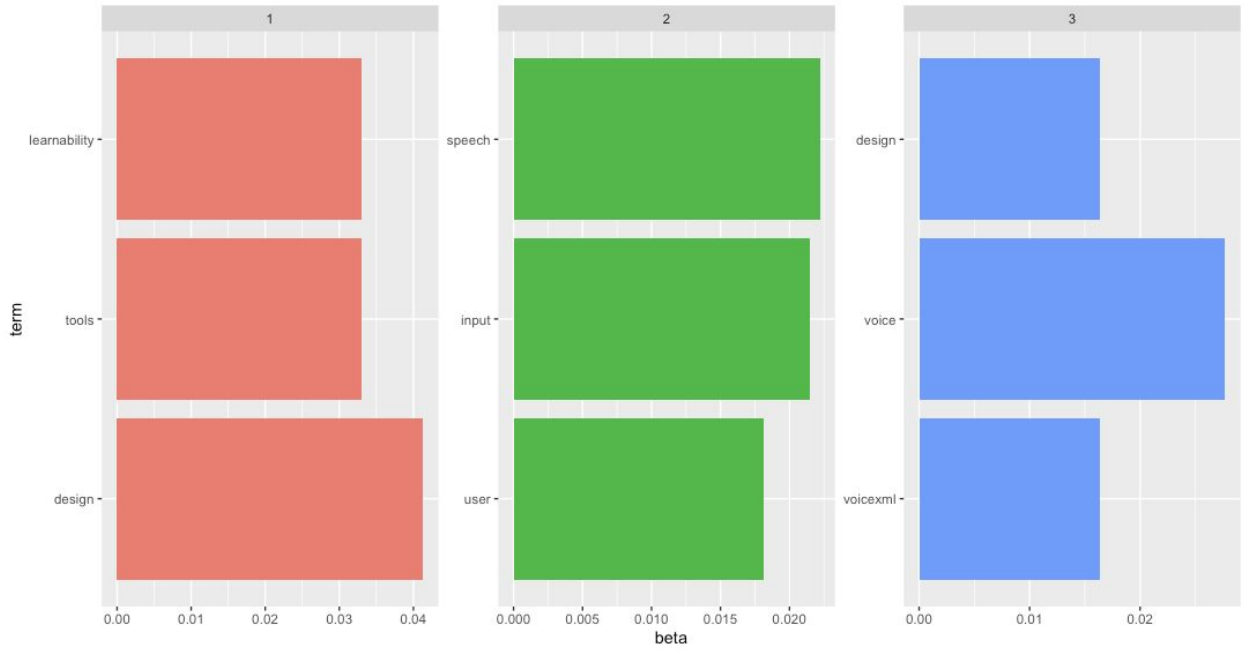


Figure 3. Topic modeling on results from “VUI” on Google Scholar.



R CODE

```
# Conversational Agents Terms

# Text Analysis | Topic Modeling


#to create and work with corpora

install.packages("tm")

YYyes#for LDA topic models

install.packages("topicmodels")

install.packages('qdap')

install.packages('textometry')

install.packages("quanteda")



library(tidytext)

library(tm)

library(stringr)

library(topicmodels)

library(tidyverse)

library(dplyr)

library(data.table)
```

```
library(textometry)

library(quanteda)


# Loading combined datasets for Medium and Google Scholar

# load your dataset and set "stringsAsFactors" to FALSE to override the fact that R will read text as a factor rather than a character

scholar_alexa <- read.csv("../DataSets/IPA_datasets/combined_gs.csv", encoding = "utf-8", header=TRUE, stringsAsFactors = FALSE)

medium_alexa <- read.csv("../DataSets/IPA_datasets/combined_m.csv", encoding = "utf-8", header=TRUE, stringsAsFactors = FALSE)


# VUIs

# load your dataset and set "stringsAsFactors" to FALSE to override the fact that R will read text as a factor rather than a character

vui_scholar_alexa <- read.csv("../DataSets/IPA_datasets/vui_gs.csv", encoding = "utf-8", header=TRUE, stringsAsFactors = FALSE)

vui_medium_alexa <- read.csv("../DataSets/IPA_datasets/vui_m.csv", encoding = "utf-8", header=TRUE, stringsAsFactors = FALSE)


# Create Corpus

# Specify the data frame source

scholar_source <- DataframeSource(scholar_alexa)

medium_source <- DataframeSource(medium_alexa)
```

```
vui_scholar_source <- DataframeSource(vui_scholar_alexa)

vui_medium_source <- DataframeSource(vui_medium_alexa)


# Transform into a corpus

scholar_corpus <- VCorpus(scholar_source)

medium_corpus <- VCorpus(medium_source)


vui_scholar_corpus <- VCorpus(vui_scholar_source)

vui_medium_corpus <- VCorpus(vui_medium_source)


# Now our text column has been transformed into a special type of format: a corpus.

print(vui_scholar_corpus)

# Using indexing we can call on a specific record. This gives metadat on that record.

print(scholar_corpus[[10]])

print(scholar_corpus[[10]][1])


# Clean the Corpus - scholar

scholar_cleaned <- tm_map(scholar_corpus, removeNumbers)

scholar_cleaned <- tm_map(scholar_cleaned, stripWhitespace)

scholar_cleaned <- tm_map(scholar_cleaned, removePunctuation)
```

```
scholar_cleaned <- tm_map(scholar_cleaned, content_transformer(tolower))


# Clean the Corpus - medium

medium_cleaned <- tm_map(medium_corpus, removeNumbers)

medium_cleaned <- tm_map(medium_cleaned, stripWhitespace)

medium_cleaned <- tm_map(medium_cleaned, removePunctuation)

medium_cleaned <- tm_map(medium_cleaned, content_transformer(tolower))


# Clean the Corpus - vui_scholar

vui_scholar_cleaned <- tm_map(vui_scholar_corpus, removeNumbers)

vui_scholar_cleaned <- tm_map(vui_scholar_cleaned, stripWhitespace)

vui_scholar_cleaned <- tm_map(vui_scholar_cleaned, removePunctuation)

vui_scholar_cleaned <- tm_map(vui_scholar_cleaned, content_transformer(tolower))


# Clean the Corpus - vui_medium

vui_medium_cleaned <- tm_map(vui_medium_corpus, removeNumbers)

vui_medium_cleaned <- tm_map(vui_medium_cleaned, stripWhitespace)

vui_medium_cleaned <- tm_map(vui_medium_cleaned, removePunctuation)

vui_medium_cleaned <- tm_map(vui_medium_cleaned, content_transformer(tolower))


#stopwords - Everbody gets the same treatment
```

```
scholar_stopwords <- c(stopwords("en"),"can", "interaction", "interact", "vui", "vuis","designed")

scholar_cleaned <- tm_map(scholar_cleaned, removeWords, scholar_stopwords)

vui_scholar_cleaned <- tm_map(vui_scholar_cleaned, removeWords, scholar_stopwords)

scholar_cleaned[[2]][1]

medium_stopwords <- c(stopwords("en"),"can", "user", "used", "users")

medium_cleaned <- tm_map(medium_cleaned, removeWords, medium_stopwords)

vui_medium_stopwords <- c(stopwords("en"),"can", "design","user", "used","“", "”", "alexa","fred",
"rogers", "interface", "the","users", "voice", "vui", "voices", "the", "like", "voic")

vui_medium_cleaned <- tm_map(vui_medium_cleaned, removeWords, vui_medium_stopwords)

vui_medium_cleaned[[2]][1]


# Stemming

scholar_cleaned <- tm_map(scholar_cleaned, stemDocument, "english")

medium_cleaned <- tm_map(medium_cleaned, stemDocument, "english")

vui_scholar_cleaned <- tm_map(vui_scholar_cleaned, stemDocument, "english")

vui_medium_cleaned <- tm_map(vui_medium_cleaned, stemDocument, "english")

vui_medium_cleaned <- tm_map(vui_medium_cleaned, removeWords, vui_medium_stopwords)


head(vui_medium_cleaned)


# Topic Modeling
```

```
# Begin by creating a document-term matrix from our clean dataset

scholar_dtm <- DocumentTermMatrix(scholar_cleaned)

medium_dtm <- DocumentTermMatrix(medium_cleaned)

vui_scholar_dtm <- DocumentTermMatrix(vui_scholar_cleaned)

vui_medium_dtm <- DocumentTermMatrix(vui_medium_cleaned)


# Get rid of empty rows, if there were any :P

unique_indexes_scholar <- unique(scholar_dtm$i)

scholar_dtm <- scholar_dtm[unique_indexes_scholar,]

scholar_dtm


unique_indexes_medium <- unique(medium_dtm$i)

medium_dtm <- medium_dtm[unique_indexes_medium,]

medium_dtm


unique_indexes_vui_scholar <- unique(vui_scholar_dtm$i)

vui_scholar_dtm <- vui_scholar_dtm[unique_indexes_vui_scholar,]

vui_scholar_dtm


unique_indexes_vui_medium <- unique(vui_medium_dtm$i)

vui_medium_dtm <- medium_dtm[unique_indexes_vui_medium,]
```

```
vui_medium_dtm


# Get it ready for Tidy! - scholar
scholar_dtm_tidy <- tidy(scholar_dtm)
scholar_dtm_tidy


cleaned_scholar_res <- scholar_dtm_tidy %>%
 group_by(document) %>%
 mutate(terms = toString(rep(term, count))) %>%
 select(document, terms) %>%
 unique()


head(cleaned_scholar_res)


# Get it ready for Tidy! - medium
medium_dtm_tidy <- tidy(medium_dtm)
medium_dtm_tidy


cleaned_medium_res <- medium_dtm_tidy %>%
 group_by(document) %>%
 mutate(terms = toString(rep(term, count))) %>%
```

```
  select(document, terms) %>%
  unique()

head(cleaned_medium_res)

# Get it ready for Tidy! - vui_scholar
vui_scholar_dtm_tidy <- tidy(vui_scholar_dtm)
vui_scholar_dtm_tidy

cleaned_vui_scholar_res <- vui_scholar_dtm_tidy %>%
  group_by(document) %>%
  mutate(terms = toString(rep(term, count))) %>%
  select(document, terms) %>%
  unique()

head(cleaned_vui_scholar_res)

# Get it ready for Tidy! - vui_medium
vui_medium_dtm_tidy <- tidy(vui_medium_dtm)
vui_medium_dtm_tidy
```

```
cleaned_vui_medium_res <- vui_medium_dtm_tidy %>%
 group_by(document) %>%
 mutate(terms = toString(rep(term, count))) %>%
 select(document, terms) %>%
 unique()

head(cleaned_vui_medium_res)



# Run LDA from Topic Modeling Package. We want 6 topics, you can choose more or less.
# We'll put 5 terms into each topic.
k <- 5

# scholar
scholar_lda <- LDA(scholar_dtm, k = k, control = list(seed=1234))
scholar_lda # A LDA_VEM topic model with 6 topics.
scholar_lda_words <- terms(scholar_lda,5)
# That gives us the top 5 terms for each of 6 topics.

# medium
medium_lda <- LDA(medium_dtm, k = k, control = list(seed=1234))
```

```
medium_lda # A LDA_VEM topic model with 6 topics.

medium_lda_words <- terms(medium_lda,5)

# That gives us the top 5 terms for each of 6 topics.



# vui_scholar

vui_scholar_lda <- LDA(vui_scholar_dtm, k = k, control = list(seed=1234))

vui_scholar_lda # A LDA_VEM topic model with 6 topics.

vui_scholar_lda_words <- terms(vui_scholar_lda,5)

# That gives us the top 5 terms for each of 6 topics.



# vui_medium

vui_medium_lda <- LDA(vui_medium_dtm, k = k, control = list(seed=1234))

vui_medium_lda # A LDA_VEM topic model with 6 topics.

vui_medium_lda_words <- terms(vui_medium_lda,5)

# That gives us the top 5 terms for each of 6 topics.




scholar_lda_topics <-as.matrix(scholar_lda_words)

head(scholar_lda_topics)

write.csv(scholar_lda_topics,file=paste("scholar_LDA_",k,".csv"))
```

```
# Use indeed_lda_words to create the *_lda_topics matrix and save as a csv.

medium_lda_topics <-as.matrix(medium_lda_words)

head(medium_lda_topics)

write.csv(medium_lda_topics,file=paste("medium_LDA_",k,".csv"))



# ---

# To visualize as a table, we will need to tidy and create a bar chart.

scholar_lda_tidy <- tidy(scholar_lda)

scholar_lda_tidy

# We now have a statistic for how much each word is associated with a topic.



# Order words from most prominent to least for each topic

top_terms <- scholar_lda_tidy %>%

 group_by(topic) %>%

 top_n(5, beta) %>%

 ungroup() %>%

 arrange(topic, -beta)

top_terms



top_terms %>%

 mutate(term = reorder(term, beta)) %>%
```

```
 ggplot(aes(term, beta, fill=factor(topic))) +
 geom_col(show.legend=FALSE) +
 facet_wrap(~ topic, scales = "free")+
 coord_flip()
# ---


# Function that does all above with any corpus
get_LDA_topics_terms_by_topic <- function(input_corpus, plot = TRUE, number_of_topics = 6,
number_of_words = 5) {
 my_dtm <- DocumentTermMatrix(input_corpus)
 unique_indexes <- unique(my_dtm$i)
 my_dtm <- my_dtm[unique_indexes,]
 my_lda <- LDA(my_dtm, k = number_of_topics, control = list(seed=1234))
 my_topics <- tidy(my_lda, matrix="beta")
 my_lda_words <- terms(my_lda, number_of_words)
 my_lda_topics <- as.matrix(my_lda_words)
 write.csv(my_lda_topics,file=paste("indeed_LDA_function",k,".csv"))
 my_top_terms <- my_topics %>%
  group_by(topic) %>%
  top_n(number_of_words, beta) %>%
  ungroup() %>%
```

```
  arrange(topic, -beta)
 if(plot==TRUE){
  my_top_terms %>%
   mutate(term = reorder(term, beta)) %>%
   ggplot(aes(term, beta, fill=factor(topic))) +
   geom_col(show.legend=FALSE) +
   facet_wrap(~ topic, scales = "free")+
   coord_flip()
 }else{
  return(my_top_terms)
 }
}
get_LDA_topics_terms_by_topic(scholar_cleaned)

# With the function we can change parameters
get_LDA_topics_terms_by_topic(scholar_cleaned, number_of_topics = 5, number_of_words = 5)
get_LDA_topics_terms_by_topic(medium_cleaned, number_of_topics = 5, number_of_words = 5)
get_LDA_topics_terms_by_topic(vui_scholar_cleaned, number_of_topics = 3, number_of_words = 3)
get_LDA_topics_terms_by_topic(vui_medium_cleaned, number_of_topics = 3, number_of_words = 3)
```

```
#--

# Rather than seeing top topics across the corpus, we would like it arranged by document

# Gamma calculation with LDA Vector


scholar_lda_document_topics <- tidy(scholar_lda, matrix="gamma")

scholar_lda_document_topics

scholar_lda_document_topics$document <- as.integer(scholar_lda_document_topics$document)


write.csv(scholar_lda_document_topics,file=paste("scholar_LDA_document_topics_",k,".csv"))

head(scholar_lda_document_topics)


# Need additional variables to analyze

dt1 <- data.table(scholar_lda_document_topics, key = "document")

dt2 <- data.table(scholar_alexa, key = "doc_id")


scholar_merged <- dt1[dt2]

dim(scholar_merged)

colnames(scholar_merged)


scholar_analyze <- select(scholar_merged, c(title))

head(scholar_analyze)
```