Universität Leipzig,
Fakultät für Mathematik und Informatik
Institut für Informatik

Titel der Arbeit: Nutzungsszenarien für Chatbots zur Verbesserung der unternehmensinternen Kommunikation

Bachelorarbeit

Leipzig, Januar 2018

vorgelegt von
Frommert, Christian
Studiengang: Informatik Bsc.

Betreuung: Dr.-Ing. Romy Elze
Institut für angewandte Informatik
Universität Leipzig

Abstract

...

Danksagung

...

Inhaltsverzeichnis

1 Einleitung

In diesem Kapitel werden Motivation, Zielsetzung und Struktur der Arbeit behandelt.

1.1 Motivation

Der Bundestagswahlkampf 2017 zeigte: Digitalisierung ist ein großes Thema. Nicht nur für die Politik, sondern auch für die Wirtschaft. Viele Unternehmen kämpfen heutzutage noch mit einer hohen Menge an Offline-Dokumenten, einige nähern sich bereits der Vision ”Papierloses Büro”, indem beispielsweise eingehende Post eingescannt und per E-Mail an den zuständigen Mitarbeiter gesandt wird und die interne Kommunikation ausschließlich auf digitalem Weg erfolgt.

Die Konsequenz daraus ist ein sehr hohes E-Mail Aufkommen, für welches laut einer internen Umfrage des IT-Service-Anbieters Atos 5 bis 20 Arbeitsstunden pro Woche pro Mitarbeiter aufgebracht werden, um die E-Mails zu lesen und zu beantworten. Als Folge bemüht man sich nun bei der internen Kommunikation auf E-Mails zu verzichten (Zero-E-Mail Policy, vgl. https://www.golem.de/1111/88078.html) und setzt verstärkt auf Enterprise Social Networks und Messaging Dienste. Das Problem hierbei ist, dass diese Dienste einfach nur weitere Tools neben den bereits vorhandenen darstellen, die jedoch nicht untereinander kommunizieren und die Mitarbeiter schließlich mit einer Vielzahl von unabhängigen Programmen arbeiten müssen.

Hier können Chatbots Abhilfe schaffen, indem sie die Informationen der einzelnen Tools bündeln und so die unternehmensinterne digitale Arbeit verbessern.

1.2 Zielsetzung

Im Rahmen der Bachelorarbeit sollen praxisnahe Szenarien gezeigt werden, in welchen Chatbots einen Teil zur Verbesserung der unternehmensinternen Kommunikation leisten. Hierfür sollen vorhandene Tools herangezogen und weiterentwickelt werden. Im Zuge der Anforderungsanalyse wird sich auf ein Enterprise Social Network bzw. einen Messaging-Dienst und einen Chatbot festgelegt. Die Bachelorarbeit soll aber so ausgelegt werden, dass die Erkenntnisse auf alle möglichen Dienste und Anwendungen übertragbar sind.

1.3 Struktur der Arbeit

Kapitel 1 stellt eine Einleitung in das Thema dar und definiert Motivation sowie Zielsetzung dieser Arbeit. Im 2. Kapitel werden die Grundlagen zu den zu behandelnden Themengebieten Enterprise Social Networks, Chatbots und Natural Language Processing erläutert. Kapitel 3 beschreibt die Use-Cases der Anwendung von Chatbots in den Enterprise Social Networks. Kapitel 4 beschreibt darauf aufbauend die Anforderungen an das Enterprise Social Network, dem Chatbot sowie den Datenquellen, um die Use-Cases umzusetzen. Im 5. Kapitel wird das Konzept der Umsetzung verdeutlicht und in Kapitel 6 die Implementierung beschrieben. Schließlich werden in Kapitel 7 die Ergebnisse zusammengefasst und ausgewertet.

2 Grundlagen

In diesem Kapitel werden die Grundlagen zum Thema dieser Bachelorarbeit behandelt. Es bietet eine Einführung in Enterprise Social Networks, Chatbots und Natural Language Processing.

2.1 Interne Unternehmenskommunikation im Jahre 2017

Wie unter 1.1 beschrieben setzen Unternehmen heutzutage verstärkt auf digitale Kommunikationskanäle.

Klassischer Weise wurde ein Unternehmens-Intranet aufgebaut, welches neben der E-Mail Kommunikation beispielsweise eine Foren-Anwendung, eine Chat-Anwendung, ein Content Management System (CMS) und ein Wiki bietet. Der Aufbau dieser Anwendungen erfolgte parallel, ohne, dass man eine Verbindung zwischen ihnen schaffen konnte. Dadurch fehlt es an einer Uniform Usability - die Mitarbeiter müssen mit vielen verschiedenen Tools arbeiten und können dabei nicht auf die Daten eines anderen Tools zurückgreifen.

Die Verwendung eines CMS zur Bereitstellung von Unternehmens-Informationen hat das Problem, dass eine sogenannte One-Way-Kommunikation stattfindet: Es gibt nur wenige Content-Creator, die meist nur aus einer Abteilung stammen. Ein Anreiz zum Beitragen eigener Informationen fehlt, die Navigation ist nicht benutzerfreundlich und die Inhalte sind oft veraltet.

War vor einigen Jahren die E-Mail noch das Non-Plus-Ultra für die Kommunikation kämpfen viele Unternehmen mit einer Vielzahl von E-Mails, die den Arbeitsalltag eher erschweren statt erleichtern. Ein weiteres Problem hierbei ist, dass die interne sowie die externen Kommunikation über einen Kanal laufen - beides landet im E-Mail Postfach. Um mal eben schnell mit dem Kollegen zu schreiben nutzen deshalb viele Angestellten ihre privaten sozialen Netzwerke - was dann jedoch die private und geschäftliche Kommunikation mischt und zu sehr ablenken kann. Deshalb nutzen immer mehr Unternehmen Enterprise Social Networks, um die interne Kommunikation zu vereinheitlichen und zu vereinfachen.

Dabei gibt es wesentliche Ziele der Einführung eines ESN. Die Kommunikation soll optimiert werden, das heißt, dass viele Mitarbeiter miteinander kommunizieren, unabhängig der Distanz zueinander. Untersuchungen zufolge ist die Frequenz der mündlichen Unterhaltungen zwischen zwei Mitarbeitern geringer je weiter die Entfernung zwischen diesen beiden Mitarbeitern ist. Der Untersuchung zufolge trifft dies auch auf die telefonische Kommunikation als auch auf die E-Mail Kommunikation zu. Schließlich kann man sagen, dass man einen Kollegen wahrscheinlich nicht anrufen oder schreiben würde wenn man ihn nicht persönlich kennt - es muss zuvor ein Informationsaustausch stattfinden. Bei E-Mails, die an die komplette Belegschaft gehen, ist das nicht möglich - in ESN ist es aber durchaus üblich Beiträge von unbekannten Personen zu sehen.

Weiteres Ziel der Einführung eines ESN ist die Steigerung der Akzeptanz. Gerade junge Angestellte nutzen privat zahlreiche soziale Netzwerke, E-Mails oder Briefe schreibt man den Freunden nicht. Warum also nicht auch im Unternehmen nutzen? Viele Arbeitgeber sehen dies als Benefit und weisen in Stellenausschreibungen darauf hin, dass im Unternehmen ESN eingesetzt werden.

Gegenüber eines CMS bietet man Anreize zum Wissensbeitrag der Angestellten. Experten können sich viel leichter einbringen.

Zuletzt sollen durch ESN Innovationen gefördert werden. Eine Untersuchung des MIT ergab, dass 80% aller Unternehmensinnovationen durch informelle Gespräche geschaffen werden. Durch den Nachrichten-Streams in ESNs, in denen Mitarbeiter beispielsweise ihre aktuelle Aufgabe preisgeben, und der Kommentarfunktion können sich schnell Ideen entwickeln, die Unternehmensprozesse vereinfachen können. Ohne diese Interaktion wäre eine solche Innovation nur schwer möglich.

2.1.1 Enterprise Social Networks

Enterprise Social Networks (ESN) bestehen normalerweise aus einen Activity Stream, in dem die Aktititäten der Nutzer einfließen, die sich in den gleichen Gruppen befinden oder die eine hohe Popularität aufweisen (z.B. durch Like-Angaben). ESN bieten eine Gruppenverwaltung, in der die Unternehmensstruktur anhand des Organigramms idealerweise abgebildet werden kann. Integrierte Anwendungen erweitern das ESN - beispielsweise um ein Umfrage-Tool, Kalender, Datei- und Dokumentenverwaltung, Projekt- und Aufgabenmanagement, Customer Relationship Management und viele weitere. Mithilfe dieser Integrationen können ggf. bislang benutzte Tools abgelöst werden.

Weiterhin sind Benutzerprofile natürlich ein zentraler Bestandteil eines ESN. In diesen können neben den Stammdaten des Mitarbeiters ein Foto hochgeladen werden, was die räumliche Nähe untereinander steigert. Überlegenswert ist es weitere Daten zu hinterlegen - beispielsweise aus dem vorhandenen Active Directory des Unternehmens oder benutzerdefinierte Felder wie Kompetenzen.

Ein weiteres Bestandteil von ESN sind die Such-, Nachrichten- und Benachrichtigungsfunktion. Kollaborationstools und Schnittstellen zu externen Tools runden das Gesamtpaket ab. ESN bieten zwar Look & Feel von bekannten privaten sozialen Netzwerken (Public Social Networks) wie Facebook und Google+, beziehen aber das gesamte Intranet des Unternehmens ein. `http://www.flyingdog.de/downloads/Enterprise_Social_Network.pdf`

2.1.2 Datenquellen im Unternehmen

Chatbots können nur intelligent arbeiten, wenn eine entsprechende Datenbasis zur Verfügung steht. Hier muss bei dem Zugriff darauf geachtet werden, dass keine Daten nach außen gelangen und die Zugriffsrechte eingehalten werden (ein Angestellter ohne Leitungsbefugnis sollte nicht auf Daten zugreifen dürfen, die Abteilungsleitern vorbehalten sind). Die Zugriffsrechte müssen entsprechend im Bot inkludiert und mit dem Rechteverwaltungssystem (z.B. Active Directory) abgeglichen werden. Bei der Anbindung an Services ist dies immer einzubeziehen.

Folgende Datenquellen stehen unternehmensintern zur Verfügung:

- **Active Directory**: Eigenschaften über Mitarbeiter, zB Geschlecht, Sitz (Büro + Niederlassung), Sprache, usw. sowie benutzerdefinierte Attribute

- **E-Mail Server**, oftmals Exchange: Zugriff auf Kalender, E-Mails
- **ERP / SAP**
- **CRM-Systeme**: Laufende Kundenprojekte, Termine, Meilensteine
- **Informationen über Produkte, Roadmaps**
- **Enterprise Social Network**: Mitarbeiter-Profile mit Kompetenzprofil, Telefonnummer, Ausbildung etc.
- **Internes Wiki**: Wissensbasis
- **Dokumentenmanagementsystem**

2.2 Chatbots

Chatbots sind NLP-basierte Programme, die je nach Benutzeraktion reagieren und so den Benutzer unterstützen. Sie bestehen aus einer Chatbot Engine, welches die Schnittstellen für die ein- und ausgehenden Nachrichten verwaltet, sowie einer Conversational Language, welche die Logik des Chatbots darstellt und zu gegebenen Nachrichten entsprechende Antworten bereitstellt. Dies ist meistens mit einem Key-Value Store verbunden, der zu Benutzern vorhandenes Wissen speichern und abrufen kann. Darüber hinaus bieten Conversational Languages NLP-Möglichkeiten in Form von regulären Ausdrücken und Substitutionen (beispielsweise steht „I'm" für „I am").

Einen guten Chatbot kann man nur entwickeln, wenn man sich vorab einen genauen Plan des Nutzens und der Interaktion macht. Anders als im unternehmens-fernen Kontext sind in der unternehmens-internen Nutzung keine generalisierten Chatbots gefragt (also keine, die allwissend sind und über alles Auskunft geben können), sondern unternehmens-spezifische, die beispielsweise eine Person mit bestimmten Kompetenzen suchen oder automatisiert aus E-Mail Inhalten Schlüsse ziehen können, um so beispielsweise einen neuen Termin im Kalender anlegen zu können. Diese und weitere Möglichkeiten werden im Kapitel „Use Cases" erläutert. Um diese Funktionen zu ermöglichen sind neben den funktionalen Definitionen aber auch die Datenquellen im Unternehmen wichtig - welche unter dem Punkt „Datenquellen" eruiert werden.

Chatbots kommunizieren mit dem Nutzer. Bisherige Lösungen in Form von Formularen oder Apps gehen einen statischen Weg (Fragen und Antwortmöglichkeiten vorgegeben). Mit Chatbots hat man die Möglichkeit einen umfassenden Datenstamm aufzubauen, der z.B.

auch auf Gespräche von vor einem Monat Bezug nehmen und so Sequenzen bauen kann, die für Formulare unmöglich abzubilden wären.

Chatbots lassen sich darüber hinaus sehr einfach in bestehende Messaging-Plattformen integrieren. Benutzer lassen sich immer weniger zum Download einer neuen App bewegen (Statistik: https://de.statista.com/statistik/daten/studie/595216/umfrage/anzahl-der-downloads-von-mobile-apps-je-smartphone-nutzer-in-deutschland/ oder auch https://qz.com/253618/most-smartphone-users-download-zero-apps-per-month/), was die Wichtigkeit darstellt, alles in einer Anwendung zu bündeln.

Durch Texterkennung lassen sich Muster in den Benutzereingaben feststellen, die nicht mit statischen Methoden feststellbar wären (zB Stimmung). Durch Texterkennung kann man - wenn man keine weiteren Daten hat - beispielsweise auch die benutzte Sprache feststellen (siehe Information Retrieval - Trigramme).

Durch Chatbots lassen sich sehr viele Prozesse automatisieren. Aktuell befinden wir uns noch in der Anfangsphase der Chatbot-Nutzung. Denkbare Szenarien sind die Benutzung von Chatbots für FAQs oder hoch standardisierte Prozesse. Der Einsatz von Chatbots ist allerdings umstritten - vor allen, wenn Daten „in die Cloud" gelangen und man die Kontrolle über die Daten verliert. http://www.faz.net/aktuell/wirtschaft/netzwirtschaft/unternehmen-setzen-auf-chatbots-chancen-risiken-14175914.html

2.2.1 Aktueller Entwicklungsstand

Aktuell gibt es unzählige Chatbots und Chatbot-Engines auf dem Markt. Zudem gibt es verschiedene Ansätze für die Modellierung des Dialogs.

Grundlegend bieten viele moderne Chatbot-Engines die Möglichkeit verschiedene Kommunikationskanäle zu bedienen und durch eigens definierte Aktionen individuelle Datenquellen heranzuziehen. Die Unterschiede zwischen den Chatbot-Engines liegen in der Usability - einige mit benutzerfreundlichen UIs, andere auf Kommandozeile -, in der Zielgruppe (PSN vs universell) sowie in Maschine Learning Aspekten.

Generell wird strikt getrennt zwischen: Eingabe, Ausgabe (was der Bot tun soll) sowie den Message-Flow. Um diese Abstraktion herzustellen wird die Eingabe vom Benutzer in sogenannte Intents und Entities übersetzt. Intents stellen hierbei die Intention des Benutzers dar, also das Themengebiet, zu dem die Benutzereingabe gehört. Der Vorteil ist, dass man verschiedene Eingaben einem Intent zuordnen kann. Beispielsweise würden die Eingaben „Ja",

„Ok" und „In Ordnung" dem Intent „Zustimmung" zugeordnet werden. Entities stellen dabei Informationsobjekte dar, die eine bestimmte Information speichern. Beispielsweise wäre bei der Benutzereingabe „Hast du morgen Zeit?" das Wort „morgen" die Entität „Zeit" mit dem Wert „morgen". Programmieraufgabe ist nun, die Entität sinnvoll zu verarbeiten - denn „morgen" stellt je nach Zeit ein anderes Datum dar. Abhilfe schaffen hier Named-entity recognition (NER) Tools dar, die flexibel eingebunden werden können. Mehr dazu im Kapitel Natural Language Processing.

2.2.2 Natural Language Processing

Durch Natural Language Processing kann man menschliche Sprache zu maschinenverständliche Anweisungen übersetzen. Hierfür werden Pruning (entfernen nicht-hilfreicher Wörter wie z.B. Stopwörter), Stemming (Grundformreduktion) und POS-Tagging angewandt, sodass am Ende ein standardisierter Satz entsteht.

Durch Intent Classification weist man diesen Sätzen dann bestimmte, vorgegebene Klassen (intents) zu. Zusätzlich kann man Entitäten (entities) bestimmten, die im vorgegebenen Satz als Variablen behandelt und später erkannt werden sollen. Den Vorgang der Entitätsbestimmung nennt man Named-entity recognition (NER). Als Beispiel der Satz „Ich suche nach einen Ort im Norden der Stadt" - intent ist hier die Ortssuche, eine Entität kann „Norden" sein als Variable für die Himmelsrichtung. Wenn später der Satz „Ich suche nach einen Ort im Süden der Stadt" analysiert werden soll wäre nun „Süden" die Entität. Das NLP-Tool liefert intent und eventuelle entitites, die Conversational Language matcht diese und liefert entsprechende Antwortsätze, die im besten Falle je nach intent und entity verschieden ausfallen.

Die Extraktion von Entities muss man entweder selbst durch Trainingsdaten anlernen - oder man verwendet NER-Tools wie Duckling, welches automatisch gängige Werte extrahieren kann (https://duckling.wit.ai/). Die erkannten Dimensionen umfassen Zeit- und Temperaturangaben, Zahlen (ausgeschrieben zu dezimal), Entfernungen, Volumen, Zeitdauer, E-Mail Adressen, Telefonnummern und URLs.

Wie im Kapitel „Chatbots" erwähnt bieten lediglich Software-as-a-Service Chatbots NLU Möglichkeiten. Da im unternehmensinternen Ansatz jedoch interne Daten als Wissensbasis verwendet werden sollen ist dies datenschutzrechtlich nicht vertretbar. Aufgrund dessen benötigen wir ein Self-Hosted Tool / on premise.

3 Konzept

Basierend auf den Grundlagen soll eine Lösung konzeptioniert werden, die man die unternehmensinterne Kommunikation durch den Einsatz von Chatbots verbessern kann.

3.1 Anforderungen

Zentrale Anforderung ist, dass die Daten das Unternehmen nicht verlassen. Es wird also auf eine Lösung Wert gelegt, die man auf einen Server des Unternehmens installieren kann und nicht nach Hause telefoniert. Hierfür ist von großer Bedeutung Open Source Tools zu verwenden, sodass man jeden Arbeitsschritt nachvollziehen und kontrollieren kann.

3.1.1 Enterprise Social Network

Der Markt gibt einige Enterprise Social Networks her - Slack, Basecamp, Asana, Atlassian Hipchat, Microsoft Teams und viele weitere. Einige davon sind allrounder, wie zum Beispiel Basecamp, andere spezialisieren sich auf ein bestimmtes Thema, wie zB Slack - Instant Messaging oder Asana - Projektmanagement, können allerdings oftmals durch Plugins um weitere Funktionen erweitert werden.

Nur die wenigsten sind uneingeschränkt kostenlos nutzbar. Ein Enterprise Social Network, welches kostenfrei nutzbar und Open Source ist, ist Humhub. Deshalb schauen wir uns im weiteren dieses soziale Netzwerk genauer an, machen uns mit den Möglichkeiten vertraut und werden dann eruieren, wie man diese Möglichkeiten nutzen kann, um die digitale Zusammenarbeit zu verbessern.

Humhub ist ein Open Source Enterprise Social Network, welches aktiv auf Github weiterentwickelt wird und durch Plugins erweitert werden kann. Die Nutzerregistrierung kann wahlweise manuell erfolgen oder man importiert eine bestehende Active Directory Infrastruktur. Anders als bei den meisten Social Networks gibt es keine Gruppen, sondern sogenannte Spaces, in denen sich Nutzer in Ihrem Interessengebiet austauschen können. Man kann individuelle Profilfelder pro Benutzer definieren, sodass jeder Benutzer beispielsweise die eigenen Kompetenzen eintragen kann. Daneben gibt es zahlreiche Module für Humhub, die das ESN um Dinge erweitert wie einen Instant Messenger, einen Kalender, ein Wiki oder ein Umfrage-Tool.

3.1.2 Chatbot

Im Zuge der Bachelorarbeit wurden die bekanntesten Chatbots recherchiert.

Name	Website	Open Source
lita	github.com/litaio/lita	X
Errbot	github.com/errbotio/errbot	X
ChatterBot	github.com/gunthercox/ChatterBot	X
ChatScript	github.com/bwilcox-1234/ChatScript	X
IKY	github.com/alfredfrancis/ai-chatbot-framework	X
Botpress	github.com/botpress/botpress	X
Rasa Core	github.com/RasaHQ/rasa_core	X
Superscript	github.com/superscriptjs/superscript	X
Bottr	github.com/Bottr-js/Bottr	X
MS Bot Framework	github.com/Microsoft/BotBuilder	X
Botkit	ithub.com/howdyai/botkit	X
E.D.D.I.	github.com/labsai/EDDI	X
Hubot	github.com/hubotio/hubot	X
Program-O	github.com/Program-O/Program-O	X
Chatfuel	chatfuel.com	-
Motion.ai	motion.ai	-
Dexter	rundexter.com	-
Pandorabots	pandorabots.com	-
Facebook Bots	messenger.fb.com	-
Rebot	rebot.me	-
IBM Watson	ibm.com/watson/services/conversation	-
Conversation Service wit.ai	wit.ai	-
Dialogflow	dialogflow.com	-

Aufgrund der aufgestellten Anforderung, dass die benutzten Lösungen on-premise gehostet werden und Open Source sein sollen, beschränken wir uns auf die oberen 14 Chatbots. Lita, Errbot, Bottr, Botkit, Hubot und Program-O besitzen keine NLU-Möglichkeiten, sodass alle Eingabesätze fest trainiert oder zumindest durch einen regulären Ausdruck hinterlegt werden müssen - ein Maschine Learning Ansatz fehlt. Die Tools könnten allerdings um NLU

erweitert werden. ChatterBot besitzt keine Abstraktion zwischen Eingabesätze und Conversation Flow, was das Training erschwert. ChatScript und SuperScript sind eher Conversation Description Languages, welche AIML verbessern möchten - die nötigen Erweiterungen zum sinnvollen Einsatz als unternehmensinternen Chatbot fehlen. IKY und E.D.D.I. fehlt es an einer umfangreichen Dokumentation. Botpress ist eher auf Facebook als primäre Zielgruppe ausgelegt. Microsoft Bot Framework ist zu sehr mit dem SaaS-Tool Cortana verknüpft, sodass zur Klassifizierung alle Eingabesätze hochgeladen werden - was nicht unseren Anforderungen genügt.

Es bleibt Rasa Core, welches von den gleichen Machern wie das NLU-Tool Rasa NLU ist. Dadurch ist eine nahtlose Integration untereinander möglich - Rasa Core empfängt den Eingabesatz, sendet es an Rasa NLU, erhält dadurch Intent und Entities und findet die zugehörige Action je nach Kontext. Zudem stehen eine gute Dokumentation, umfangreiche Plugin-Möglichkeiten und eine große Community dahinter.

3.2 Use Cases

Aufgrund der in Kapitel 2.1.2 genannten Datenquellen lassen sich viele verschiedene Use-Cases definieren, die stets als Ziel verfolgen, ein externes Tool bzw. die Datenquelle aus dem Messaging-Dienst heraus zu beeinflussen.

So könnte man beispielsweise alle E-Mails in Exchange nach einen Wort oder mehreren Worten durchsuchen und den Inhalt einer gefundenen E-Mail zurückgeben (sofern Exchange die Schnittstelle dafür bereitstellt).

Im Zusammenhang mit einen ERP-System könnte man bei der Lieferung von Inventar automatisiert eine Nachricht über den Inhalt der Lieferung verfassen.

Ein anderer Use-Case im Zusammenhang mit dem ERP-System wäre ein Request-Approval-Modell für die Anfrage von bestimmten Equipment sowie Approval-Prozess über die entsprechende Person - alles über den Messaging-Dienst statt dem ERP-System.

Bei CRM Systemen könnte man Aktualisierungen von Kundenprojekten an alle betroffenen Personen im Messaging-Dienst weiterleiten, so wie man dies bei Integrationen von beispielsweise Github in Slack bereits kennt: Hier können alle Github-Neuigkeiten wie Pull Requests oder neue Issues zu einen Repository in einen Slack-Channel kommuniziert werden. Die Nutzer können daraufhin die Änderung kommentieren und so einen neuen Inhalt zum CRM hinzufügen.

Aber auch ohne eine Datenquelle lassen sich Use-Cases finden, die den Unternehmensalltag erleichtern. Beispielsweise hat die Firma avocado einen Chatbot für ein internes Event der Firma Zalando konzipiert, der die Angestellten vor und während der Veranstaltung unterstützte, indem Informationen zum Veranstaltungsort, zum Ablauf hinsichtlich der Vorträge und zu weiteren Dingen bereitgestellt wurden - beispielsweise das WLAN-Passwort, Standort der Toiletten und Fragen zur Verpflegung. Falls der Chatbot keine Antwort zur Frage geben konnte wurde automatisch ein menschlicher Mitarbeiter zu Rate gezogen.

Das Ergebnis: Es gab 466 Anfragen an den Bot mit einer Erfolgsquote von 98%. Ein menschlicher Mitarbeiter musste nie zu Rate gezogen werden (siehe https://chatbotsmagazine.com/how-we-got-a-98-success-rate-for-our-bot-for-zalando-a-ux-case-study-fcdc0e70469d). Der Vorteil gegenüber einer eigenen Website oder App: Die Benutzer können die Infos schnell erhalten, die sie wirklich benötigen (statt auf der Website oder in der App zu suchen), zudem können sie direkt im gleichen Programm mit den Kollegen chatten. Bei Verwendung einer eigenen App nur für das Event müsste man sich erst die App installieren - dies entfällt beim Einsatz eines Chatbots.

Um den Rahmen dieser Arbeit nicht zu übersteigen sollen die beiden folgenden Use Cases umgesetzt werden.

3.2.1 Kalender

Der Aufwand, einen freien Termin unter mehreren Kollegen zu finden, soll durch diesen Bot minimiert werden. Aktuell würde eine Terminfindung zwischen Mitarbeiter A und Mitarbeiter B so verlaufen, dass A in seinen Kalender schaut, freie Termine für die nächsten zwei Tage sucht und diese B vorschlägt. B würde daraufhin seinen Kalender öffnen und schauen, ob die vorgeschlagenen Termine ebenfalls frei sind. Wenn nicht würde er an einen anderen Tag suchen und der Ball würde wieder bei A liegen.

Diesen Prozess könnte man mit einer einfachen Abfrage auf beide Kalender stark vereinfachen. Ein Chatbot könnte auf beide Kalender zugreifen und freie Termine matchen. Im freien Markt gibt es hierfür bereits Tools und Apps, die jedoch wieder ein neues Tool nur für diese eine Verwendung darstellen. Der Vorteil des Chatbots liegt darin, dass man direkt im Messaging-Dienst den freien Termin finden und einen Kalendereintrag hierfür erstellen kann.

In der freien Wirtschaft wird für die Termin- und Raumbuchung überwiegend Microsoft

Exchange benutzt, auf das in dieser Bachelorarbeit jedoch nicht zurückgegriffen werden kann. Bei einer Umsetzung mit Microsoft Exchange könnte man neben den freien Terminen in den Kalendern zusätzlich einen freien Raum für diese Zeit suchen oder den Status eines spezifischen Raumes abfragen und in den Terminvorschlag einfließen lassen. Denkbar wäre auch die Arbeitszeiten zu parameterisieren.

3.2.2 Kompetenzen

Die Suche nach Personen im Unternehmen mit bestimmten Kompetenzen gestaltet sich umso schwieriger, je mehr Personen im Unternehmen arbeiten, obwohl die Anzahl der Kompetenzen steigt. Das Resultat ist, dass Aufgaben unzureichend erledigt werden oder unbearbeitet bleiben, nur weil man keine Kenntnis der Kompetenzen der Mitarbeiter und Kollegen besitzt. Aktuell kann man nur durch Zufall herausfinden, dass der Kollege aus der anderen Abteilung die Kompetenz besitzt, die man benötigt.

Wenn man die Benutzerprofile von Humhub um Kompetenzen erweitert kann man Schlüsselwörter aus Chatnachrichten extrahieren und diese den geforderten Kompetenzen zuordnen, um so schnell eine geeignete Person zu finden. Der Vorteil des Einsatzes des Chatbots gegenüber der manuellen Variante (im Unternehmen Durchfragen, E-Mail an alle Angestellten, etc.) liegt hier klar in der Geschwindigkeit sowie der globalen Suche.

In den Unternehmen wird überwiegend Microsoft Active Directory eingesetzt, mit welchen man benutzerspezifische Daten hinterlegen kann, wie beispielsweise Kompetenzen, hinterlegen kann. Man könnte also statt aus Humhub die Daten direkt aus dem Active Directory abfragen. Weiterhin ist überlegenswert, ob man den Online-Status der Person in Betracht zieht und so nur Nutzer vorschlägt, die online sind.

3.3 Konzept

3.3.1 Middleware

Bevor man die Use-Cases umsetzen kann müssen zuerst Enterprise Social Network, Chatbot und NLU verbunden werden.

Inhalte müssen aus dem ESN zum NLU gegeben werden, um intent und entities zu klassifizieren. Intent und Entities sollen dann an den Chatbot gegeben werden, um eine Antwort

bzw. Reaktion zu finden. Schließlich soll das Ergebnis wieder in das ESN fließen. Rasa-Core bietet bereits eine Integration für Rasa-NLU, sodass wie uns nur noch um die Verbindung zwischen Humhub und Rasa-Core kümmern müssen.

Da das Mail-Modul von Humhub keine API mitbringt müssen wir die Datenbank mit den Nachrichten monitoren, neue Nachrichten automatisch an Rasa-Core weiterleiten und die Antworten von Rasa-Core in der Datenbank speichern. Deswegen benötigen wir eine Middleware, die diese Aufgaben erledigt. Die Nachrichten müssen an eine Sender-ID geknüpft werden, sodass eine Zuordnung der Antwortsätze auf die richtige Konversation erfolgen kann. Eine wichtige Anforderung hierbei ist, dass die Datenübermittlung ausschließlich innerhalb des Unternehmens passiert, also keine Daten ins Internet gesendet werden müssen (beispielsweise für die Klassifikation). Außerdem sollte der Chatbot als Middleware-Lösung mit verschiedenen Enterprise Social Networks bzw. auch Instant Messaging Services und NLU-Tools kompatibel sein, da NLU-Tools meist nur wenige Sprachen unterstützen und man so für andere Szenarien andere NLU-Tools einsetzen muss.

3.3.2 Chatbot

Der Chatbot muss um die Funktionalität erweitert werden mit der Middleware zu kommunizieren und als Reaktion auf bestimmte Eingabesätze Unternehmensressourcen heranzuziehen, wie beispielsweise den Kalender oder eine Kompetenz-Matrix. Da die Aktion und der Antwortsatz direkt im Chatbot generiert wird ist dies Aufgabe des Chatbots statt der Middleware. Außerdem müssen Beispielsätze für die Use-Cases gefunden, mit Intents und Entities verknüpft und trainiert werden. Schließlich müssen verschiedene Dialog-Abläufe gefunden werden, die möglichst viele Fälle abdecken. Für die nicht abgedeckten Fälle ist es nötig eine sogenannte Policy zu definieren, die für diese Abweichungen die Aktion findet, die am nächsten zur Situation passt.

Essentiell wichtig ist die richtige Zuordnung zu den Gesprächspartnern, sodass die Datensicherheit gewährleistet wird. Hierfür muss die Middleware eine Sender-ID weitergeben und Rasa Core diese mit der Antwortnachricht zurücksenden.

4 Implementierung

4.1 Rasahub

Es wurde ein Tool umgesetzt, welches sich mit der Humhub-Datenbank verbindet, die ID der neuesten Nachricht abfragt, speichert und in einer Endlosschleife jeweils wieder die neuste ID abfragt und vergleicht. Wird nun eine ID gefunden, die von der gespeicherten abweicht, wird die Chat-Nachricht der ID abgefragt und per Socket-Connection an den RasahubInput-Channel gegeben. Hierfür muss eine Server-Socket-Verbindung eingerichetet werden, womit sich RasahubInputChannel als Client verbinden kann.

Rasahub kann direkt von den pypi Quellen installiert werden:

pip install rasahub

Es wird automatisch Rasa_Core, Rasa_NLU und ein MySQL-Connector installiert.

Folgende Parameter sind nun zu berücksichtigen:

Parameter	**Langform**	**Beschreibung**	**Erforderlich**	**Standardwert**
-dbu	–dbuser	Datenbank Benutzername	X	
-dbp	–dbpwd	Datenbank Benutzerpasswort	X	
-dbh	–dbhost	Datenbank Hostadresse	-	127.0.0.1
-dbn	–dbname	Datenbank Name	X	
-t	–trigger	Trigger Zeichenkette	-	!bot
-id	–botid	(Humhub) Benutzer-ID	X	
-rh	–rasahost	Rasa Hostadresse	-	127.0.0.1
-rp	–rasaport	Rasa Port	-	5020

Rasahub kann direkt von der Kommendozeile gestartet werden. Ein beispielhafter Aufruf könnte so aussehen:

rasahub -dbu humuser -dbp secretpassword -dbn humhub -id 5

Nun wird auf eine Verbindung vom RasahubInputChannel gewartet:

Connected to database. Waiting for socket connection from Rasa on port 5020

4.2 RasahubInputChannel

RasahubInputChannel verbindet sich per Socket-Verbindung zu Rasahub und empfängt und sendet darüber Nachrichten zum und vom Chatbot. Ein InputChannel hat nach Maßgabe von Rasa-Core die Klassen InputChannel und OutputChannel zu definieren, wobei der Input-Channel auf den OutputChannel in der Methode _record_messages zugreift. Weiterhin muss der Inputchannel die Methoden __init__, start_async_listening und start_sync_listening implementieren. In der _record_messages Methode wird die Nachricht der Socket-Verbindung empfangen. Im Outputchannel ist neben __init__ die Methode send_text_message nötig, in der die Antwort von Rasa-Hub an die Socket-Verbindung gesendet wird.

Sobald der InputChannel über das run.py Skript initialisiert wurde verbindet sich Rasa_Core mit Rasahub:

Connection established: ('127.0.0.1', 47794)

Bei jeder neuen Nachricht in Humhub erhält man Debug-Informationen in Rasahub:

Input from db: u'message': u'Ich habe ein Problem mit meiner Tastatur', u'message_id': 4 Reply from rasa: u'reply': u'Christian Frommert koennte bei dem Anliegen helfen.', u'message_id': 4

4.3 Chatbot

4.3.1 Trainingsdaten

4.3.2 Domain

5 Evaluation

6 Zusammenfassung

Literatur

[1] https://www.xovi.de/2017/05/chatbots-die-vor-und-nachteile-automatisierter-kommunikation/

[2] https://de.wikipedia.org/wiki/Ein-Prozent-Regel_(Internet)

[3] http://www.bvdw.org/presseserver/SocialMedia/kompass_social_media_2016_2017.pdf

[4] https://www.computerwoche.de/a/enterprise-social-network-im-alltags-check,3329548

[5] http://www.flyingdog.de/downloads/Enterprise_Social_Network.pdf

[6] https://convcomp.it/conversational-vs-transactional-chatbots-172166f3d2fc

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann.

Ort, Datum, Unterschrift