B.6 Kollaborative Erstellung von intelligenten Mentoring-Bots als skalierbare Werkzeuge zur individuellen Unterstützung in der Hochschulbildung

Research

Alexander Tobias Neumann[1], *Ralf Klamma*[1]
[1] *RWTH Aachen University, Informatik, Informationssysteme & Datenbanken*

1 Einführung und Motivation

Heutzutage sind virtuelle persönliche Assistenten allgegenwärtig. Assistenten wie Alexa, Siri und Google Assistant werden sowohl für einfache Aufgaben wie die Bereitstellung des Wetterberichts als auch für personalisierte Inhalte, wie z.B. den nächsten geplanten Termin, verwendet (Dale, 2016; Dibitonto, Leszczynska, Tazzi & Medaglia, 2018; Yan, Castro, Cheng & Ishakian, 2016). Menschen kommunizieren oft täglich mit ihnen, sodass ihre Unterstützung im Alltag vieler Nutzer bereits tief verwurzelt ist. Neben diesen allgemeinen Assistenten gibt es auch spezialisierte Versionen, die auf die Lösung domänenspezifischer Aufgaben zugeschnitten sind. Die große Mehrheit davon sind domänenspezifische Chatbots (Dale, 2016). Diese Technologien sind weit verbreitet, da Facebook beispielsweise eine eigene Programmierschnittstelle (API) zur Verfügung stellt, mit der Chatbots erstellt werden können, die meist von Prominenten oder Unternehmen genutzt werden und häufig gestellte Fragen automatisch beantworten können (Ferrara, Varol, Davis, Menczer & Flammini, 2016). Die rasche Zunahme ihrer Popularität hat auch zu Kontroversen geführt, insbesondere beim Einsatz in sozialen Medien (Ferrara et al., 2016; Shao et al., 2018). Der Einsatz von sozialen Bots in intelligenten Tutorensystemen, insbesondere im Kontext von MOOCs, ist eine neuere Entwicklung. In Bildungskontexten werden Bots bereits im medizinischen Bereich, beim Sprachenlernen und zur metakognitiven Unterstützung eingesetzt. In informellen Lernsituationen wie dem Mentoring werden soziale Bots jedoch nur selten, wenn überhaupt, eingesetzt. An Universitäten, an denen Vorlesungen von mehreren hundert Studenten besucht werden, ist eine Eins-zu-eins-Betreuung durch einen Lehrenden mit persönlichem Feedback kaum machbar. Die Entwicklung eines Bots für eine bestimmte Webanwendung, der direkt mit dem Nutzer zusammenarbeitet, ist eine Aufgabe, die ein tiefes technisches Verständnis sowohl für die Anwendung als auch für die Entwicklung von sozialen Bots erfordert. Anforderungen, die insbesondere Praxengemeinschaften (CoP) (Wenger, 1998) oft nicht erfüllen. Eine CoP ist eine Gruppe von Menschen, die durch gemeinsames Fachwissen und Leidenschaft für eine gemeinsame Praxis verbunden sind. Der Einsatz von sozialen Bots in diesen Gemeinschaften bietet ein Maß an Lernunterstützung, das ansonsten aufgrund des Mangels an verfügbaren Ressourcen nicht möglich wäre (Roda, Angehrn, Nabeth & Razmerita, 2003). Die modellgetriebene Entwicklung birgt das Potenzial, die Abstraktionsebene zu erhöhen und ermöglicht dadurch eine bessere Integration der Endnutzer in die Entwicklungsprozesse.

Unser Beitrag stellt eine webbasierten modellgesteuerten Umgebung für soziale Bots vor mit der wir CoPs die Möglichkeit geben, ihre eigenen Lernassistenten zu erstellen. Unser Ansatz kann verwendet werden, um soziale Bots für die eigene Organisation zu erstellen und auf Webdienste zurückgreifen kann, indem man sich auf RESTful APIs mit OpenAPI-Spezifikation verlässt, um die Aktionen des sozialen Bots innerhalb der Anwendung zu definieren. Abschließend werden drei Beispiele von Mentoring Bots vorgestellt, die mit dem Framework erstellt wurden und zeigen das große Potential der Anwendung. Mit den Bots erhalten Studierenden personalisiertes Feedback, ohne dass die Lehrkräfte zusätzlich belastet werden.

2 Theoretischer Hintergrund & Begleitende Arbeiten

2.1 Soziale Bots

Ein Bot ist ein automatisiertes Computerprogramm, das in der Lage ist, sich wiederholende Aufgaben zu übernehmen, die normalerweise von Menschen ausgeführt werden. Der Hauptvorteil von Bots ist, dass sie keine direkte Interaktion mit einem Menschen benötigen, um bestimmte Aufgaben zu erfüllen. Die ersten Ansätze von Social Bots wurden in den 1960er Jahren entwickelt, als Weizenbaum mit seinem Bot Eliza demonstrierte, dass es möglich ist, auf Benutzereingaben zu reagieren und Gespräche in Textform zu führen (Weizenbaum, 1966). Das Computerprogramm wird also durch bestimmte Schlüsselwörter ausgelöst, die in der Anfrage des Benutzers vorkommen. Weitere Forschungen führten zur Gründung der XML-kompatiblen Sprache AIML (Wallace, 2003). Sie besteht aus Kategorien, die einen Stimulus, also ein Muster, und eine Vorlage für die Antwort enthalten. Die kategorialen Muster innerhalb des Stimulus werden abgeglichen, um die beste Antwort auf die Eingabe des Benutzers zu erzeugen. Das Potenzial von Bots wurde früh erkannt, und verschiedene Dialogsysteme wurden mit AIML angepasst, um Anwendungen in den Bereichen Medizin und Bildung zu verbessern (Mikic, Burguillo, Llamas, Rodriguez & Rodriguez, 2009; van Rosmalen, Eikelboom, Bloemers, van Winzum & Spronck, 2012). Chatbots die vordefinierte Regeln zur Beantwortung von Benutzeranfragen verwenden werden als „abrufbasierte" Bots bezeichnet (Babar, Lapouchnian & Eric Siu-Kwong Yu, 2017). Im Gegensatz dazu nennt man Bots, die in der Lage sind sich an den Benutzer anzupassen und natürliche Sprache verstehen, als „generative" Bots (Babar et al., 2017). Generative Inhalte hängen vom aktuellen Kontext ab und berücksichtigen den jüngsten Verlauf zwischen dem Bot und dem Benutzer. Mithilfe von Techniken des maschinellen Lernens lernen die Bots aus dem Verlauf und können auf dieser Grundlage beliebige Fragen beantworten (Serban et al., 2017; Varghese & Pillai, 2018). Die Einbeziehung von nicht-technischen Nutzern kann zum Beispiel mit SOCIO erfolgen (Perez-Soler, Guerra & Lara, 2018). Dabei handelt es sich um ein System, bei dem Benutzer kollaborative Anfragen stellen können und das System nach jeder Änderung das entsprechende Klassendiagramm zurückgibt. Die Zusammenarbeit der Teilnehmer und die Erfassung der Anforderungen findet in einem Messenger statt.

2.2 Bots als Mentoren

MOOCs sind eine Weiterentwicklung traditioneller Kurse, die vollständig online angeboten werden und ein weitaus größeres Publikum ansprechen. Dieses große Publikum hat seinen Preis: Persönliche Interaktionen zwischen Studierenden und Lehrkräften müssen geopfert werden, um diese Kurse lebensfähig zu machen. Da es bei dieser Art von Kursen keine Anwesenheit vor Ort gibt, können die Studierenden leicht isoliert werden. Der Einsatz von Bots in einem solchen Kontext besteht aus drei Hauptzielen:

- Bereitstellung eines 24/7-Service für Studierende, der ohne Bots teuer oder in einigen Fällen schlichtweg unmöglich wäre (Lim & Goh, 2016).
- Die Arbeitsbelastung menschlicher Lehrkräfte verringern, indem man Bots häufig gestellte Fragen beantworten lässt.
- Studenten helfen, mit anderen Kommilitonen oder Lehrern in Kontakt zu treten und Gespräche zwischen ihnen zu fördern (Bozkurt, Kilgore & Crosslin, 2018).

MOOC-Bot ist ein Chatbot für eine MOOC-Website mit dem Zweck, häufig gestellte Fragen zum Kurs zu beantworten, Wissen über den Kurs zu vermitteln und eine Chatbot-Schnittstelle bereitzustellen (Lim & Goh, 2016). In unserem Beitrag legen wir den Schwerpunkt auf die Lösung der oben genannten Ziele, indem wir eine Modellierungsumgebung anstelle von Skriptsprachen (z. B. AIML) verwenden, um die Erstellung für nicht-technische Benutzer zu erleichtern. Teacherbot ist ein Twitter-Bot für den MOOC „E-Learning and Digital Cultures" auf der Coursera-Plattform (Bozkurt et al., 2018). Die Rolle des Bots ist es, Diskussionen zwischen Studierenden zu starten und Fragen zur Organisation des Kurses zu beantworten. Der Bot twittert in regelmäßigen Abständen Zitate zum Kursinhalt und antwortet auf Tweets, die einen bestimmten Hashtag enthalten. Er zieht andere Studierende oder Lehrende in die Diskussion mit ein, um die Schüchternheit zu überwinden, denn er folgt der Idee, dass es für schüchterne Menschen einfacher ist, ein Gespräch mit einem Bot zu beginnen als mit einem anderen Menschen. Der Link Student Assistant (LiSA) ist ein Chatbot in Form eines virtuellen Assistenten, der Studenten bei allgemeinen Aufgaben auf dem Universitätscampus hilft (Dibitonto et al., 2018). Der Bot wurde eingesetzt, um die Studierenden per Chat zu ihren Anforderungen an einen Bot zu befragen und um herauszufinden, wie sich die „Persönlichkeit" des Bots auf die Benutzererfahrung auswirkt. Hier wurde der Chatbot über Googles Dialogflow-Framework implementiert und über den Facebook Messenger angesprochen. Eicher et al. zeigen mit Jill Watson, dass auch sensible Informationen ausgetauscht werden können, sie betonen den Vorteil der Nutzung eines privaten und vertraulichen Kanals (Eicher, Polepeddi & Goel, 2018). Daher legen wir in unserer Arbeit großen Wert auf Datenschutz und verwenden nur Open-Source-Software, die von uns selbst gehostet werden kann.

3 Social Bot Framework

3.1 Architektur

Die Architektur des Social Bot Frameworks (Neumann, de Lange & Klamma, 2019) ist in drei Teile unterteilt (siehe Abbildung 1: Architekturübersicht des Social Bot Frameworks.). Für die Interaktion mit Studierenden kommen hauptsächlich Messenger wie Rocket.Chat, Slack oder Telegram zum Einsatz. Der Kern des Frameworks ist der Manager Service[1]. Für die Integration der Chat-Komponenten stellen wir im Manager Service eine abstrakte Klasse bereit und haben die Entwicklung weitegehend modular und möglichst einfach gehalten, damit in Zukunft weitere Chat-Dienste eingebunden werden können. Eingehende Nachrichten können über eine kontinuierliche WebSocket-Verbindung empfangen werden. Nachdem sich der Bot erfolgreich bei dem Messenger angemeldet hat, lauscht er auf eingehende Nachrichten. Die Intent-Klassifizierung und die Chat-Verbindung werden in der Modellierungsansicht als ein Element zusammengefasst. Die eingehenden Nachrichten werden an einen Rasa[2] NLU-Server gesendet, um die Intention und Entitäten einer Chat Nachricht abzurufen. Die Definition von Intentionen und die entsprechenden Beispielanweisungen können im nutzerfreundlichen Markdown-Format angegeben werden. Der Manager Service verwaltet die erstellten Bot Modelle und NLU Server. Die Bot Modelle können über das bereitgestellte Frontend (siehe Kapitel 3.2) erstellt werden. Neben der Chat Kommunikation bietet das Bot Framework die Möglichkeit RESTful Microservices, die den OpenAPI[3] Standard unterstützen, anzusprechen.

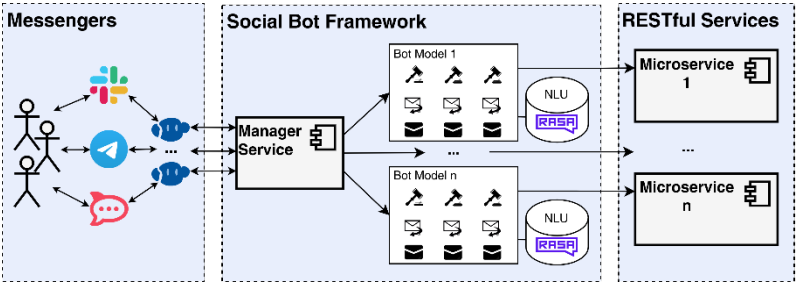

Abbildung 1: Architekturübersicht des Social Bot Frameworks.

[1] https://github.com/rwth-acis/las2peer-social-bot-manager-service

[2] https://rasa.com

[3] https://www.openapis.org

3.2 Bedienoberfläche

Die Bedienoberfläche[4] des Social Bot Frameworks wurde als Webanwendung entwickelt und liegt Open Source zur Verfügung. Abbildung 2: Weboberfläche des Social Bot Frameworks. bietet einen Überblick über unsere Modellierungsumgebung, welche auf einer kollaborativen (Meta-)Modellierung basiert, die nahezu in Echtzeit arbeitet (Nicolaescu, Rosenstengel, Derntl, Klamma & Jarke, 2018). Im oberen Bereich hat der Endnutzer die Möglichkeit das aktuelle Modell an das Backend zu schicken und den Bot zu starten, das Modell zu speichern oder ein bereits verfügbares zu laden. Der Modellierungs-Canvas (links) zeigt das aktuelle Modell, während die Palette (mittig) die verfügbaren Modellierungselemente enthält. Zusätzlich ermöglicht das Attributswidget (rechts) die Änderung der Attribute der Elemente oder ihrer Verbindungen.

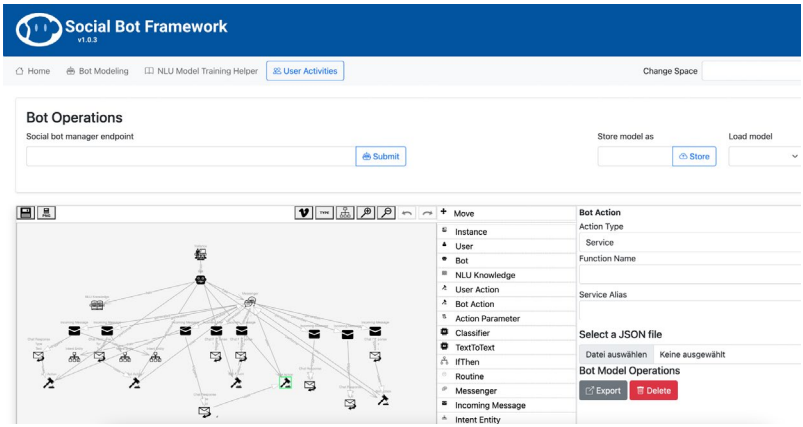

Abbildung 2: Weboberfläche des Social Bot Frameworks.

Für ein funktionales Botmodell ist es wichtig, die Funktionen, auf die der Bot reagieren soll oder den Bot auslösen sollen, korrekt zu modellieren. Bei den ausführbaren Aktionen wird zwischen Bot- und Nutzeraktionen unterschieden. Eine *Benutzeraktion* bezieht sich auf eine vom Nutzer ausgeführte Funktion eines RESTful Dienstes, wobei eine *Botaktion* eine vom Bot ausgeführte Aktion darstellt. *Botaktionen* werden durch einen Auslöser mittels Instruktionsregeln, wie zum Beispiel einer *Routine* oder einer *Benutzeraktion*, initiiert. Eine *Routine* ist ein zeitgesteuerter Auslöser (z. B. ein Wochentag oder/und eine bestimmte Uhrzeit). Um den Bot als Chatbot einzusetzen, muss dieser mit einem *Messenger* verknüpft werden. Hierfür muss ein Bot-Token oder eine andere vom *Messenger* verlangte Authentifizierungsinformationen angegeben werden, sodass der Bot Zugriff zum Kommunikationsmedium bekommt.

[4] https://github.com/rwth-acis/Social-Bot-Framework

Anschließend können Nachrichten, die der Bot bewältigen soll modelliert werden. Die Erkennung der Nachrichten findet mittels der Rasa NLU Komponente statt (siehe Kapitel 3.1). Somit lassen sich mühelos FAQ Bots erstellen.

3.3 Mentoring Bots

Im Folgenden stellen wir drei Bots vor, die mit dem Framework modelliert wurden und im Rahmen des tech4comp[5] Projekts für Mentoring oder generell zur Unterstützung in der Hochschulbildung eingesetzt werden.

AssessmentBot. Mit dem Assessment Bot (Neumann, Conrardy & Klamma, 2021) wurde geprüft, wie Moodle-Quizze automatisch in Dialogsysteme integriert werden können. Ein RESTful Microservice wurde genutzt um Bots Assessments mit Nutzern durchführen können. Diese umfassen Quizzes, die mithilfe der bereitgestellten RESTful-API aus der Moodle-Plattform extrahiert werden. Nach dem Absolvieren eines Moodle-Quiz innerhalb des Chats werden die Daten des absolvierten Quiz an ein LRS gesendet. Dabei stellten wir fest, dass sowohl der Kommunikationsaspekt als auch die Mobilität der Chat-Plattform für das mobile Self-Assessment sprechen. Ein Aspekt, der während der Evaluierung zur Sprache kam, war die Kombination der Quizdaten mit anderen Daten, da der LRS die Aggregation von Daten ermöglicht. Ein Beispiel wäre die Verwendung von Stresssensoren, um den Zusammenhang zwischen dem Absolvieren von Quizzes und Stress zu analysieren.

FeedBot. Im Zusammenhang mit Mentoring in der Hochschulbildung wird Feedback als ein wesentliches Instrument hervorgehoben, um die Fähigkeiten und den Wissensstand der Studierenden sichtbar zu machen und so den Studierenden zu helfen, ihren eigenen Lernprozess zu überwachen. Eines der Hauptziele von FeedBot (Neumann, Arndt et al., 2021) ist es, individuelles Feedback auf skalierbare Weise zu geben. Dazu wurde ein computergestütztes linguistisches Analysetool (T-MITOCAR) verwendet, um automatisch Repräsentationen von Wissen aus Prosatexten zu erstellen. Das Feedback aus der Analyse der Studierenden und einem Vergleich der Wissensstruktur aus dem Schülertext mit den entsprechenden Wissensmodellen aus der Seminarliteratur zusammen.

LitBot. Mit LitBot (Neumann, Arndt et al., 2021) werden Studierende mit Übungen über Lektüren begleitet und so zum Selbststudium motiviert und strukturiert. Beim Lesen von Sachtexten müssen die Leserinnen und Leser auf ihr Vorwissen zurückgreifen. Vorwissen und domänenspezifisches Wissen beeinflussen und tragen zum Verständnis eines Textes und zum Lernen aus einem Text bei.

[5] https://tech4comp.de

Die Aktivierung von Vorwissen vor dem Lesen kann helfen, relevante Informationen im Text zu identifizieren und vernetztes Lernen zu ermöglichen. LitBot ist so konzipiert, dass es Übungen zur Aktivierung des Vorwissens bereitstellt, bevor ein Schüler den Text gelesen hat. Darüber hinaus bietet LitBot nach der Lektüre Aufforderungen zum Nachdenken über den Text und Empfehlungen für weiterführendes Material, das auf den spezifischen Interessen der SchülerInnen in Bezug auf den Lesestoff basiert.

4 Zusammenfassung und Ausblick

In diesem Beitrag haben wir ein webbasiertes, modellgesteuertes Framework zur Erstellung von Social Bots für RESTful-Webanwendungen vorgestellt. Mit deutschen Intent-Erkennungsmodellen wurden Bots im Rahmen eines bundesweiten Forschungsprojekts erstellt und auf verschiedene Anwendungsszenarien umgesetzt. Die erstellten Bots geben zu jeder Tageszeit Antworten auf einfache FAQ und sind in der Lage, direktes Feedback auf Seminaraufgaben zu geben. Während der Laufzeit des Projekts werden die Bots in dieser ausgedehnten Nutzungsphase seit mehreren Semestern und auch kommenden Semestern evaluiert und verbessert. Wir sind davon überzeugt, dass Mentoring Bots in Learning Management Systemen ihren alltäglichen Lebensraum finden werden. In Zukunft werden wir auch neue Bots für weitere Szenarien anbieten.

Literatur

Babar, Z., Lapouchnian, A. & Eric Siu-Kwong Yu (2017). Chatbot Design – Towards a Social Analysis Using i* and process architecture. In iStar Workshop, S. 73-78.

Bozkurt, A., Kilgore, W. & Crosslin, M. (2018). Bot-teachers in hybrid massive open online courses (MOOCs): A post-humanist experience. Australasian Journal of Educational Technology, pp. 39-59. https://doi.org/10.14742/ajet.3273

Dale, R. (2016). The return of the chatbots. Natural Language Engineering, 22(05), pp. 811-817. https://doi.org/10.1017/S1351324916000243

Dibitonto, M., Leszczynska, K., Tazzi, F. & Medaglia, C. M. (2018). Chatbot in a Campus Environment: Design of LiSA, a Virtual Assistant to Help Students in Their University Life. In M. Kurosu (Ed.), Human-Computer Interaction. Interaction Technologies. 20th International Conference, HCI International 2018, Las Vegas, NV, USA, July 15–20, 2018, Proceedings, Part III (Lecture Notes in Computer Science, vol. 10903, pp. 103-116). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-91250-9_9

Eicher, B., Polepeddi, L. & Goel, A. (2018). Jill Watson Doesn't Care If You're Pregnant: Grounding AI Ethics in Empirical Studies. In Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society (AIES '18, pp. 88-94). New York, NY, USA: Association for Computing Machinery

(ACM).
Ferrara, E., Varol, O., Davis, C., Menczer, F. & Flammini, A. (2016). The rise of social bots. Communications of the ACM, 59(7), pp. 96-104. https://doi.org/10.1145/2818717
Lim, S. L. & Goh, O. S. (2016, 26. Januar). Intelligent Conversational Bot for Massive Online Open Courses (MOOCs).
Mikic, F. A., Burguillo, J. C., Llamas, M., Rodriguez, D. A. & Rodriguez, E. (2009). CHARLIE: An AIML-based chatterbot which works as an interface among INES and humans. In J.-V. Benlloch-Dualde (ed.), 2009 EAEEIE annual conference. Valencia, Spain, 22 - 24 June 2009 (pp. 1-6). Piscataway, NJ: IEEE.
Neumann, A. T., Arndt, T., Köbis, L., Meissner, R., Martin, A., Lange, P. de et al. (2021). Chatbots as a Tool to Scale Mentoring Processes: Individually Supporting Self-Study in Higher Education. Frontiers in Artificial Intelligence, 4, pp. 64-71. https://doi.org/10.3389/frai.2021.668220
Neumann, A. T., Conrardy, A. D. & Klamma, R. (2021). Supplemental Mobile Learner Support Through Moodle-Independent Assessment Bots. In W. Zhou & Y. Mu (Hrsg.), Advances in Web-Based Learning – ICWL 2021 (Lecture Notes in Computer Science, Bd. 13103, Bd. 13103, pp. 75-89). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-90785-3_7
Neumann, A. T., de Lange, P. & Klamma, R. (2019). Collaborative Creation and Training of Social Bots in Learning Communities. In 2019 IEEE 5th International Conference on Collaboration and Internet Computing (CIC) (pp. 11-19). IEEE.
Nicolaescu, P., Rosenstengel, M., Derntl, M., Klamma, R. & Jarke, M. (2018). Near Real-Time Collaborative Modeling for View-Based Web Information Systems Engineering. Information Systems, 74, pp. 23-39. Verfügbar unter: http://www.sciencedirect.com/science/article/pii/S0306437916305269
Perez-Soler, S., Guerra, E. & Lara, J. de. (2018). Collaborative Modeling and Group Decision Making Using Chatbots in Social Networks. IEEE Software, 35(6), S. 48-54. https://doi.org/10.1109/MS.2018.290101511
Roda, C., Angehrn, A., Nabeth, T. & Razmerita, L. (2003). Using conversational agents to support the adoption of knowledge sharing practices. Interacting with computers, 15(1), pp. 57-89.
Serban, I. V., Sankar, C., Germain, M., Zhang, S., Lin, Z., Subramanian, S. et al. (2017). A Deep Reinforcement Learning Chatbot. Verfügbar unter: http://arxiv.org/pdf/1709.02349v2
Shao, C., Ciampaglia, G. L., Varol, O., Yang, K.-C., Flammini, A. & Menczer, F. (2018). The spread of low-credibility content by social bots. Nature Communications, 9(1), 4787. https://doi.org/10.1038/s41467-018-06930-7

Van Rosmalen, P., Eikelboom, J., Bloemers, E., van Winzum, K. & Spronck, P. (2012). Towards a game-chatbot: extending the interaction in serious games. Proceedings of the 6th European Conference on Games Based Learning.

Varghese, E. & Pillai, M. T. R. (2018). A Standalone Generative Conversational Interface Using Deep Learning. In ICICCT-2018. Proceedings of the International Conference on Inventive Communication and Computational Technologies (ICICCT 2018) : 20-21 April, 2018 (pp. 1915-1920) [Piscataway, New Jersey]: IEEE.

Wallace, R. (2003). The Elements of AIML style. Alice AI Foundation.

Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. Communications of the ACM, 9(1), pp. 36-45. https://doi.org/10.1145/365153.365168

Wenger, E. (1998). Communities of Practice. Learning, Meaning, and Identity (Learning in doing). Cambridge, UK: Cambridge University Press.

Yan, M., Castro, P., Cheng, P. & Ishakian, V. (2016). Building a Chatbot with Serverless Computing. In D. Bermbach, D. Eyers & E. Wittern (Hrsg.), Proceedings of the 1st International Workshop on Mashups of Things and APIs. MOTA '16 (pp. 1-4).