PalmQA: Ein Frage-Antwort-Ensemble für selbstgesteuertes Lernen

Falk Pollok[1], Ulrik Schroeder[1], Mohamed Amine Chatti[1]

Abstract: Eine große Herausforderung für selbstgesteuertes Lernen ist das selbständige Finden und Bewerten geeigneter Lernmaterialien. Zur Lernerunterstützung können Empfehlungssysteme genutzt werden, die die populärsten Materialien aufzeigen. Dabei werden weniger populäre Nischeninhalte per definitionem übersehen. Wir stellen ein Question-Answering-Ensemble vor, das als Ergänzung zu Empfehlungen und Filtern fungieren kann, indem es semantisches Verständnis nutzt, um Antworten zu natürlichsprachlich formulierten Fragen zu extrahieren. Durch die Kombination verschiedener quelloffener Systeme und Korpora konnte die Antwortrate gegenüber aktuellen vergleichbaren Einzelsystemen verdoppelt und die Antwortzeit um zwei Drittel verkürzt werden. Das System wurde in drei Nutzerstudien evaluiert, wobei sich sowohl breite Nutzerakzeptanz zeigte als auch im Quizwettbewerb zu Faktenfragen gegenüber 26 menschlichen Mitstreitern eine überlegene Leistung erzielt wurde. Wir diskutieren die Integration des Prototyps in Lernmanagementsysteme und stellen eine GUI-Erweiterung als Messaging-Dienst vor, die Themen interaktiv erschließbar macht.

Keywords: Question Answering, e-Learning, Cognitive Computing, Keyphrase Extraction, Distributional Semantics

1 Einleitung

Von Emanuel Goldbergs Statistischer Maschine und Vannevar Bushs Memex über Kollaborationssysteme wie Ted Nelsons Projekt Xanadu, Douglas Engelbarts oN-Line System (NLS) und Tim-Berners Lees ENQUIRE bis hin zu Apples Knowledge-Navigator-Konzept von 1987 ist die Geschichte der Informatik eng mit dem Thema von Wissenszugang verquickt, welches heute mit Persönlichen Assistenten wie Amazon Echo, Google Assistant, Microsoft Cortana und Apples Siri in greifbare Nähe gerückt ist. Eine neue Schlüsselrolle spielen dabei Question-Answering-Systeme (QA) wie IBMs DeepQA, das durch den Sieg gegen menschliche Grand Champions in Jeopardy 2011 breite Bekanntheit erlangt hat. Es gehört zum Paradigma des Cognitive Computing, in dessen Rahmen diverse Machine-Learning- und Neuro-Linguistische-Programmier-Techniken (NLP) genutzt werden, um menschlicher agierende Systeme zu entwickeln.

Die in diesem Artikel dargestellte Entwicklung, detailliert in [Pol16], zielt darauf ab, QA-Systeme für e-Learning und Forschung nutzbar zu machen. Sie knüpft u.a. an Piagets Beobachtung an, dass Kognition ein konstruktiver Prozess ist. QA-Systeme haben das

[1] Informatik 9, Learning Technologies, RWTH Aachen University Ahornstraße 55, 52074 Aachen, Germany, falk.pollok@rwth-aachen.de, [schroeder,chatti]@informatik.rwth-aachen.de

Potenzial, Lernen als individuelle Wissensschöpfung zu unterstützen, da sie Lernenden ermöglichen, ein Feld sowie die Zusammenhänge zwischen Konzepten für sich zu erkunden.

Ein Problem des selbstgesteuerten, informellen Lernens ist die eigenständige Auswahl der Lernmaterialien, weshalb Filter und Empfehlungssysteme als mögliche Unterstützung zum Einsatz kommen. QA-Systeme können eine dritte Säule in diesem Modell bilden, da sie große Textcorpora bearbeiten, relevante Passagen oder Antworten extrahieren und ihre Wahl semantisch begründen können. Ferner können sie den „Long Tail“ erschließen: Es gibt wenige sehr populäre Titel, den „Short Head“, sowie eine lange Reihe von Nischeninhalten, die durch Suchmaschinen und Mainstream-Medien kaum erschlossen werden. QA-Systeme sind nicht auf Popularitätsmetriken angewiesen, sondern agieren auf Basis natürlicher Sprachverarbeitung. Daher können sie helfen, sich durch Popularität hartnäckig haltende Fehlinterpretationen aufzulösen. Ferner sind QA-Systeme nur auf Text begrenzt, sondern funktionieren prinzipiell auch auf anderen Medien wie Bildern und Videos, was sie umso vielseitiger für den Bildungseinsatz macht.

1.1 Forschungsfragen

In der hier beschriebenen Arbeit wurde ein Prototyp eines Ensembles aus QA-Systemen für die Integration in Lernumgebungen entwickelt. Dabei mussten vor allem praktische Herausforderungen gelöst werden, um passable Antwortzeiten und -raten für die Unterstützung während Lernphasen zu erzielen und nachzuweisen, wie sich neue Domänen abdecken lassen. Auf dieser Basis sollten mögliche Mehrwerte von QA-Systemen für Forschung und Lehre erhoben werden. Mit dem Prototyp wurde untersucht, wie sich die Interaktion dynamisch gestalten und die Begrenzung auf Faktenfragen einschränken lässt.

2 Stand der Forschung

QA-Systeme erhalten eine Frage in natürlicher Sprache und ermitteln eine Antwort. Sie können entweder domänenoffen oder -spezifisch sein und sind häufig auf Faktenfragen beschränkt. Frühe Systeme wie Baseball [Gr61] sind nahezu ausschließlich domänenspezifisch. Natürlichsprachliche Datenbankschnittstellen wie LUNAR [Wo73] bieten den Vorteil, dass Nutzer Anfragesprachen und Struktur einer Datenbank nicht kennen müssen, um Anfragen zu stellen. In den 70ern kommen Wissensbasen und Expertensysteme auf, üblicherweise von Domänenexperten von Hand entwickelt, die eine weite Palette an Entscheidungsunterstützung abdecken. Während diese Zeit viele neue Konzepte wie Logikprogrammierung und regelbasierte Systeme hervorbringt, gelingt der Durchbruch nicht. Auch symbolische Ansätze zum Sprachverständnis bringen zwar Lösungen wie PoS-Tagging und Named Entity Recognition hervor, heben die Stagnation aber nicht auf, bis 1993 mit MURAX [Ku99] das Interesse wieder aufflammt, welches Information Retrieval mit natürlicher Sprachverarbeitung koppelt, was insbesondere im

Kontext von Mobilgeräten und Suchmaschinen Vorteile bieten kann. 1999 wird die Text Retrieval Conference TREC etabliert. TREC und das ACQUAINT-Programm bilden ein Fundament für IBM's PIQUANT [Pr03], auf welchem wiederum die spätere DeepQA-Architektur fußt, die mit Watsons Jeopardy-Sieg weite Aufmerksamkeit erfährt.

Charakteristisch für DeepQA [Fe10], welches auf der NLP-Architektur UIMA [Fe09] aufbaut, ist die Kombination von strukturierten und unstrukturierten Wissensquellen. Das zentrale Element ist eine so genannte Common Analysis Structure (CAS), welche das zu analysierende Artefakt ebenso beinhaltet wie alle Metadaten, die von Analyse-Komponenten generiert werden. Es existiert also eine gemeinsame Datenstruktur, in der alle Analyseergebnisse zentral erfasst werden können, so dass alle Komponenten auf den für sie relevanten Daten arbeiten können. Die Pipelines aus Lese-, Analyse- und Annotationsschritten können sowohl sequentiell als auch (via BPEL) nicht-sequentiell sein und verteilt sowie parallelisiert ausgeführt werden.

Um eine Antwort zu finden, extrahiert DeepQA zunächst Schlüsselphrasen aus der Frage sowie den Fokus - jenen Teil der Frage, der die Antwort repräsentiert und aus dem der Lexikalische Antworttyp (LAT) abgeleitet wird. Da keine Ontologie ausdrucksstark genug war, ist der LAT ein Bag of Words, also ein Multiset natürlicher Sprachausdrücke. Anschließend wird die Frage klassifiziert, was die nachfolgende Pipeline bestimmt. Zum Suchen von Antwortkandidaten werden drei Strategien verwendet: Title-Oriented Documents (TODs) wie Wikis, Textpassagensuche und strukturierte Daten. Für jeden Kandidaten wird die Quelle aus der er stammt hinterlegt. DeepQA nutzt nahezu 500 Scoring-Strategien, die via logistischer Regression in einem Konfidenzwert gebündelt werden, der mit der Antwort ausgegeben wird. Das offene System YodaQA [BŠ15] ist stark an diese Architektur angelehnt, beschränkt sich aber auf TODs, Fakten und Mainstream-Suchverfahren. Dennoch gehört es mit einer selbst angegebenen Akkuranz von 32.6% zu den besten offenen Systemen.

Im Rahmen des Educational QA (EQA) haben Wen et al. [We12] bereits QA-Systeme in LMS vorbereitet, sind allerdings in Antwortrate und -zeit durch ihr (Einzel-)Basissystem OpenEphyra limitiert. Gurevych et al. [Gu09] haben Community-QA-Seiten wie Yahoo Answers für Bildungszwecke nutzbar gemacht und sind im Ansatz, Fragen zu matchen, ähnlich zu Wang et al. [Wa06], die im Rahmen von Collaborative Learning FAQ-Kataloge abfragbar und dynamisch erweiterbar gemacht haben. Atapattu et al. [AFF15] haben EQA-Systeme erweitert, um automatisch Concept Maps zu generieren und so die strukturelle Bildung von Wissen zu unterstützen.

3 Innovationen

Der Prototyp PalmQA wurde als Microservice-Architektur (vgl. Abb. 1) realisiert, die verschiedene QA-Systeme einbindet, um die Antwortgenauigkeit für den Einsatz im eLearning deutlich zu steigern.

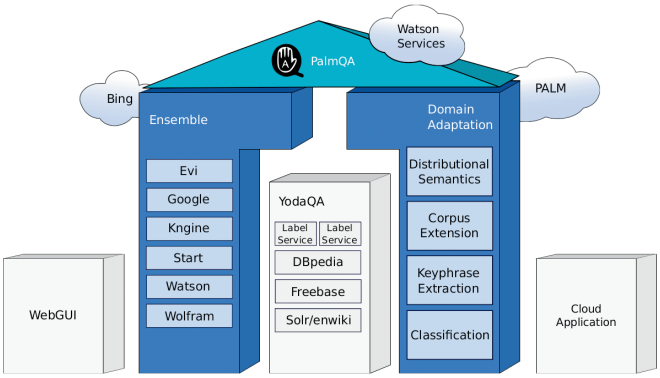

Abb. 1: Übersicht über die Microservice-Architektur

Das erste Ensemble bestand aus Amazon's Evi, Google, Kngine, MIT's Start, IBM's Watson QA API (eingestellt) sowie Wolfram Alpha. Später wurde auch Bings QA hinzugefügt. QAKiS wurde nicht mit aufgenommen, da es eine zu schlechte Präzision aufweist und seine Hauptquelle DBpedia bereits hinreichend durch YodaQA abgedeckt wird. Neben dem Ensemble wurden zwei graphische Nutzerschnittstellen für Inhaltsproduzenten und -konsumenten implementiert, ein Webinterface für die Integration in andere Systeme, ein Classifier, um vordefinierte FAQs abzufangen, eine neue Messaging-Metapher sowie Werkzeuge, um den Korpus zu erweitern, Schlüsselphrasen zu extrahieren und Distributional Semantics zu nutzen. Das Ensemble wurde erweitert, um neben QA auch auf generischen Backends wie den beiden letztgenannten zu funktionieren. Zudem wurden IBMs Clouddienste und PaaS Bluemix eingesetzt, was zu einem hybriden Clouddesign führte. Die Software kann dabei sowohl in der Cloud als auch on-premise ausgeführt werden, wobei für jeden Bluemixdienst eine On-Premise-Fallbacklösung bereitsteht.

3.1 Frage-Antwort-Ensemble

Das Ensemble unterstützt Question Answering, die Extraktion von Schlüsselphrasen sowie Distributional Semantics, alle in Dockercontainern gekapselt, sowohl für Antwortlisten (selektiert in Nutzerstudie) als auch Einzelausgaben (quantitative Analyse).

Question Answering

Um zusätzliche QA-Backends hinzuzufügen, wurden APIs für Watson und Wolfram benutzt, wohingegen für alle anderen Quellen AJAX-Parsing via HTMLUnit zum Einsatz kam. Während der Pilotstudie wurden alle Nutzerinteraktionen zusammen mit den Systemantworten via Hibernate in eine H2-Datenbank geloggt, woraus eine Menge von 185 Anfragen hervorging, die dann benutzt wurde, um die Anzahl korrekter und inkorrekter Fragen sowie Enthaltungen zu ermitteln, aus denen schließlich Precision und Recall für jedes Backend abgeleitet werden konnte. Während das Ensemble komplexere Filter anbietet wie Thresholding von Konfidenzwerten oder Mischstrategien für Listen, etwa nach Mehrheitskriterien, liefert bereits die einfache Heuristik, Quellen nach absteigender Präzision auf dem Testdatensatz abzugehen und die erste gelieferte Antwort zu akzeptieren, sehr gute Ergebnisse. Diese Strategie hat zudem den Vorteil, Einzelantworten statt Listen zu liefern und nicht alle Quellen abfragen zu müssen.

Nachträgliches Ranking ist konzeptuell leicht integrierbar, derzeit aber nicht implementiert. Während kurz versucht wurde, Antwortkandidaten nach paarweisen Ähnlichkeitsmetriken (Block, Chapman, Cosine, Dice, Euclidean, Jaccard, Jaro, Jaro-Winkler, Levenshtein, Monge-Elkan, Needleman-Wunch, QGrams, Smith-Waterman, Soundex, Tag-Link) sowie dem längsten gemeinsamen Substring zu vereinen, war dies bisher nicht erfolgreich - vermutlich, da diese Strategie im Gegensatz zu Distributional Semantics semantische Ähnlichkeit außer Acht lässt. Da die mit der Heuristik erzielten Werte angenähert den optimalen entsprechen, wurde der Forschungsschwerpunkt zunächst auf andere Themen gelegt.

FAQ-Klassifikator Da viele Fragen zu komplex für aktuelle QA-Systeme sind und Lehrkräfte Antworten erweitern können sollen, werden häufige Fragen abgefangen und vordefiniert beantwortet. Um diese Funktionalität zu testen, wurden 100 Fragen aus zwei e-Learning-Vorlesungen sowie einem Seminar manuell extrahiert. Anschließend wurde jede Frage mehrfach paraphrasiert, um verschiedene Formulierungen im Datensatz abzubilden und dann einen Classifier darauf zu trainieren - online mit IBMs Natural Language Classifier, der auf CNNs basiert, und offline mit Random Forests sowie SVMs in Weka. Diese Umformulierung können in einem LMS durch Studierende oder z.B. via Crowdsourcing Plattformen wie Mechanical Turk erfolgen, um sowohl abgedeckte Themenanzahl als auch Genauigkeit zu verbessern. Für die eigene Klassifizierung wurden Fragen zunächst in Wortvektoren überführt und dann jedes Wort nach seiner Häufigkeit in der ersten Drittel Millionen von Häufigkeiten im Google Web Trillion Word Corpus sowie der Übereinstimmung mit Schlüsselphrasen aus dem Lehrstuhlkorpus gewichtet. Um Wekas Attribute-Relation File Format effektiv zu generieren, kam ein PATRICIA-Trie zum Einsatz, der es erlaubt, Worte nach Präfixen nachzuschlagen, wobei nur ein neuer Knoten und eine neue Kante hinzugefügt werden müssen, um einen neuen Eintrag abzubilden.

Korpus-Import Um Wissensbasis und Knowledge-Graphen erweitern zu können, müssen neue Dokumente importiert werden können. Der entsprechende Dienst in Bluemix ist allerdings nur bis 100MB kostenfrei. Während er sehr saubere Ausgaben liefert und sogar Formeln in ASCII-Repräsentationen überführt, wurde der Lehrstuhlkorpus mit mehreren Gigabyte an e-Learning-Artikeln lokal mit Apache Tika konvertiert. Zusätzlich wurde der LAK-Datensatz mit eigenem Code umgewandelt und Daniel Schneider von der Universität von Genf hat einen XML-Dump seines Edutech-Wikis zur Verfügung gestellt, welches nach manueller Säuberung in Solrs XML-Format gebracht wurde. Allerdings können die zugrundeliegenden Ontologien nicht auf domänenspezifische Anfragen angewendet werden und die Dokumentart wechselt von Wikis zu Textpassagen, was in einem so signifikanten Leistungseinbruch resultiert, dass dieses Thema für sich in zukünftigen Projekten erforscht werden muss.

Schlüsselphrasenextraktion

Die Schlüsselphrasenextraktion wurde sowohl mit Rapid Automatic Keyword Extraction (RAKE), welches hierfür auf Python 3 portiert wurde, als auch mit IBMs Alchemy API

durchgeführt. Das Ensemble kann je nach Wunsch der Anwendung beide Quellen parallel abfragen und nur von beiden gefundene Schlüsselphrasen akzeptieren, was einerseits die Qualität erhöht, andererseits aber auch gültige Einträge übersieht.

Distributional Semantics

Die durch den Wechsel von Jeopardy zu Fachdomänen beobachtete signifikante Leistungsverschlechterung wurde durch den Einsatz von Distributional Semantics durch das JoBim-Text-Framework [Bi13] realisiert. Der Ansatz folgt der Auffassung, dass die Bedeutung jedes Wortes vollständig durch die umgebenden Wörter abgeleitet werden kann. Dr. Martin Riedl (TUD) hat freundlicher Weise sein existierendes Wikipediamodell zur Verfügung gestellt und zusätzlich einen Distributional Thesaurus (DT) für den Lehrstuhlkorpus berechnet, da der lokale Aachener Hadoopcluster nicht genug Speicher dafür hatte. Mit diesem DT war es möglich, via JBT Worte zu clustern, Hearstmuster zu extrahieren und schließlich gelabelte Sinncluster zu berechnen, die in einer MariaDB-Instanz abgelegt in Echtzeit abgefragt werden können, was einen ersten Schritt hin zur Unterstützung domänenspezifischer Corpora darstellt.

Der praktische Nutzen ist, dass hiermit eine lexikalische Expansion durchgeführt werden kann, also etwa das Wort „exceptionally“ zu den sinnverwandten Begriffen „extremely, extraordinarily, incredibly, exceedingly, remarkably“ erweitert werden konnte, was für die erweiterte Betrachtung von LATs nützlich ist, wo jeder Kandidat in einem eigenen Thread betrachtet werden kann. Zusätzlich zu JBT verfügt das Ensemble nunmehr über die Word-Embedding-Ansätze word2vec und GloVe, um eine umfassende Grundlage für zukünftige Forschung zu bieten. Darüber hinaus können Word-Embeddings einfacher und ohne große Hadoopcluster berechnet werden.

Containerisierung

Um diese zahlreichen Microservices handhaben zu können, wurde jeder in ein eigenes Dockerimage gebündelt, welche gemeinsam mit Docker Compose orchestriert werden können. YodaQA etwa benötigt DBpedia, Freebase, enwiki in Solr sowie zwei Stichwortdienste und einen Container für sich selbst. Da Freebase und DBpedia auf derselben Technologie (Apache Fuseki) beruhen, können beide Container aus einem gemeinsamen Image gestartet werden. Dasselbe gilt für die Stichwortdienste. Welche Ordner als Volume gemountet werden sowie der genaue Befehl und die Umgebungsvariablen entscheiden, welchen „Phänotyp“ ein Image zur Laufzeit annimmt. Die Containerisierung wurde in das offizielle YodaQA-Repository aufgenommen.

Weitere Images sind das Ensemble, ein Backend, um alle webbasierten QA-Systeme anzusprechen, eines für Bluemix-Dienste, eines für die RAKE-Schlüssel-phrasenextraktion sowie drei für Distributional Semantics: JBT zusammen mit seiner MariaDB-Datenbank in einem dedizierten Image sowie ein gemeinsames Abbild für beide Word-Embeddings-Ansätze. Um bei Erweiterungen Aufwand zu sparen, existieren generische Backends, die neben dem üblichen Kommunikationsweg über REST-Schnittstellen prinzipiell auch einen Thriftmodus anbieten, der etwa zum Einsatz kommt, um das Ensemble an die Plattform für Intelligente Assistenten Lucida anzubinden.

3.2 User Experience

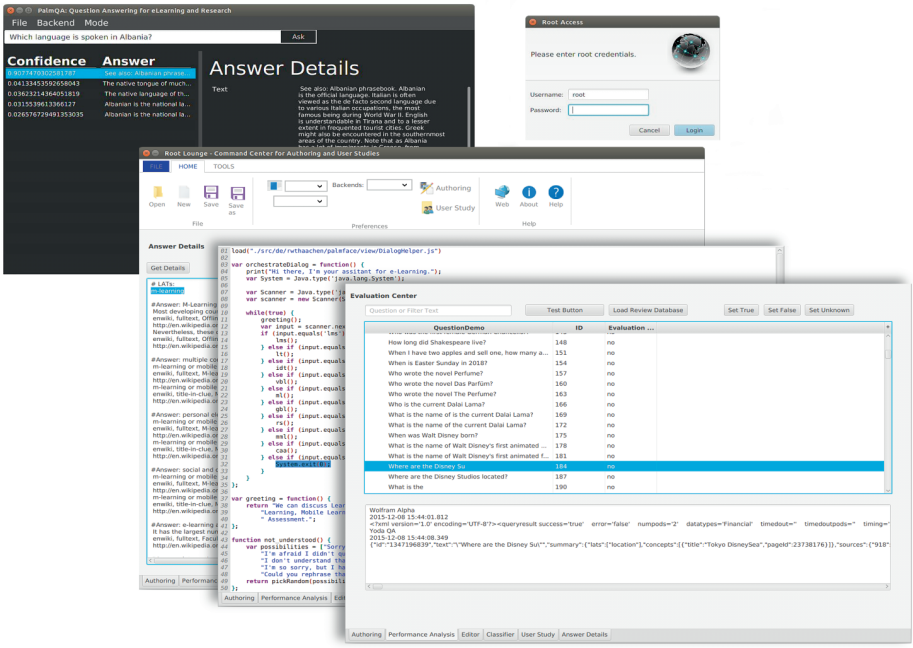

Abb. 2: Übersicht über die verschiedenen PalmQA-Benutzungsschnittstellen

Die GUI für Inhaltskonsumenten (Abb. 2) besteht aus vier Modi: Hauptmenü, QA, Dialogkomponente und eingebettetem Browser, um Zusatzquellen wie Wikipedia-Artikel, DBpedia-Einträge oder Bingtreffer sowie Wolframs Inhalte (insbesondere interaktive Pro-Features) anzuzeigen. Jene für Inhaltsproduzenten dient der Analyse von Antworten und Zusatzaufgaben wie dem Skripten von Dialogsequenzen.

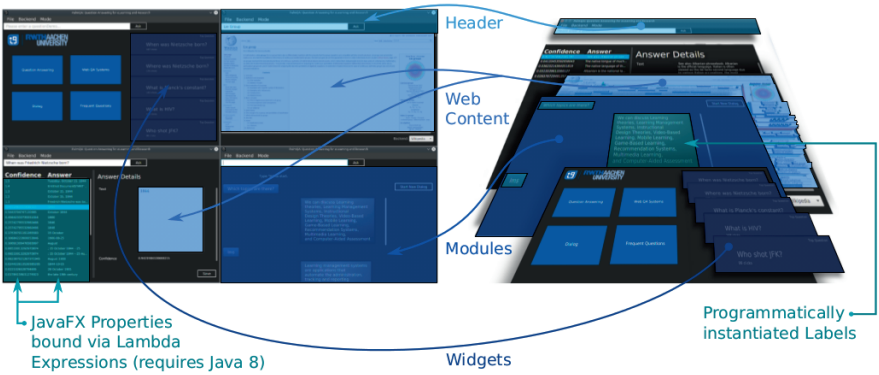

Abb. 3: GUI Modularisierung

Genauso wie die Backends als entkoppelte Microservices abgebildet wurden, ist die Benutzeroberfläche ebenfalls hochgradig modular (vgl. Abb. 3): Die Kopfzeile ist unabhängig von den auswechselbaren Hauptmodulen, sowohl Antworten als auch ihre Belege werden in WebViews eingebettet dargestellt, Listen sind JavaFX-Properties, die durch Lambda-Ausdrücke an das entsprechende Feld gebunden wurden und Dialogfelder ebenso wie FAQ-Widgets auf der Menüseite werden programmatisch generiert. Jede GUI-Komponente ist in FXML definiert und verfügt über einen eigenen Controller, wobei komplexe Interaktionen und Kommunikation untereinander über einen Hauptcontroller abgewickelt werden.

Messaging-Metaphorik

Um die Interaktion gegenüber herkömmlichen QA-Systemen dynamischer zu gestalten, wurde als Nebenprojekt eine Messaging-Metaphorik entwickelt (Abb. 4), die einerseits QA-Systeme in einer an Instant-Messaging angelehnten Kontaktliste darstellt und andererseits einfache, geskriptete Dialoge anbietet, um Themen interaktiv erkunden zu können. Hierzu werden im Grußtext Oberthemen genannt, die dann mit Folgefragen erschlossen werden können. Diese Metapher ist keine Kernkomponente und daher nicht als vollständiges Dialogsystem angelegt, sondern dient als Demonstration, um in den Nutzerstudien ein Vorabbild zukünftiger Erweiterungen erfragen zu können.

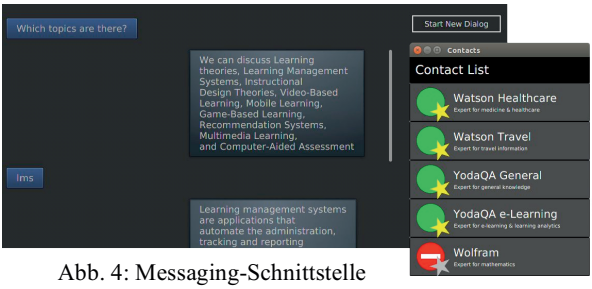

Abb. 4: Messaging-Schnittstelle

Web-GUI und Integration

Das auf der Adressbuchdemo von Vaadin basierende Webinterface, kann als iFrame eingebunden werden. Auf dieser Grundlage wurde das System mit der RWTH Lehr- und Lernplattform L²P integriert. In der Nachfolge erfolgte die Integration mit der Plattform für Intelligente Assistenten Lucida, die an der University of Michigan entwickelt wird, wodurch das mitgelieferte QA-System OpenEphyra durch das in Geschwindigkeit und Antwortrate überlegene PalmQA-Ensemble ersetzt werden kann. Das Projekt hat das Ensemble mit in sein Repository aufgenommen.

4 Evaluation

4.1 Quantitative Evaluation

Zunächst sei die Antwortgeschwindigkeit auf den später detaillierten „Grand-Challenge-Fragen“, welche viele Fragekategorien abdecken, grob analysiert: YodaQA war in der

Standardkonfiguration mit nicht-lokalen Datenquellen am langsamsten (Durchschnitt 23.9s, Beschleunigungsfaktor 1, [min. 5s .. max. 64s]), etwas schneller in der offiziellen Demo auf ailao.eu (19.8, 1.21, 2..51) und bereits doppelt so schnell, wenn die Datenquellen ebenfalls lokal ausgeführt werden (12.03, 1.98). YodaQA zeigte dabei auffällig hohe IO-Raten, was darauf schließen lässt, dass IO eher Flaschenhals als reine CPU-Kraft ist. Diese Beobachtung bestätigt sich beim Einsatz einer einfachen Consumer-SSD (8.53, 2.8, 1..25), die nahezu dreifache Beschleunigung liefert. Interessanter Weise verbessern weder größere SSDs noch eine professionelle Samsung 950 Pro mit M.2 NVMe über PCIe 3.0 mit deutlich besserem Datendurchsatz (1582MB/s statt 269MB/s für vorige 850 Evo) diesen Wert erheblich. Selbst mit Freebase auf der 850 Evo und allen anderen Backends auf der 950 Pro bleibt die Geschwindigkeit nahezu stabil (8.3, 2.88, 1..19) mit leichter Verbesserung für den ungünstigsten Fall. Die Leistungseinbußen durch Dockercontainer sind mit etwa 7% vertretbar (9.53, 2.51, 2..28) vs. (10.2, 2.34, 2..29) mit allen Backends auf 850 Evo SSD). Bei einer Extremfallmessung mit jeder Quelle auf einer dedizierten SSD hat sich die Geschwindigkeit sogar im Vergleich zu nur einer SSD verschlechtert (10.2, 2.34, 1..29). Auch Arbeitsspeicher mit 32GB gegenüber 16GB (8.6, 2.78, 1..27) zu (9.23, 2.59, 2..26) hat einen messbaren, aber nicht erheblichen Einfluss, weshalb festzuhalten ist, dass einzig die Nutzung einer lokalen SSD eine maßgebliche Verbesserung einbringt, wohingegen alle anderen Parameter verblassen.

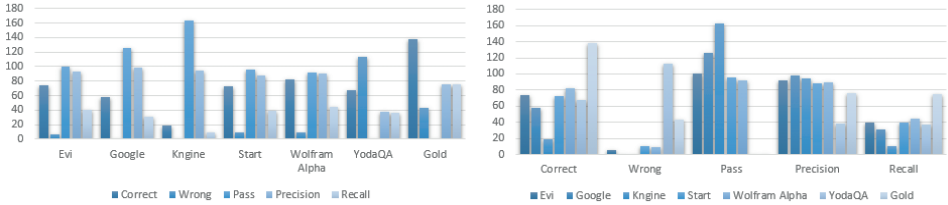

Abb. 5: Übersicht über Quantitative Evaluation

Um einen Eindruck der Ensemblekomponenten zu erhalten, wurden diese gegen die 185 während der Nutzerstudien gesammelten faktoiden Fragen getestet (vgl. Abb. 5). Hieraus ergibt sich die folgende Ordnung nach Präzision, wobei die Tupel die Anzahl korrekter Antworten, dann teilkorrekter Antworten, inkorrekter Antworten, Enthaltungen sowie Präzision und Recall wiedergeben: Google (58, 0, 1, 126, 0.98, 0.31), Kngine (19, 1, 1, 163, 0.95, 0.10), Evi (74, 5, 6, 100, 0.925, 0.4), Wolfram Alpha (82, 2, 9, 92, 0.90, 0.44) und Start (73, 5, 10, 96, 0.88, 0.39). Wie sich zeigt, weisen alle Backends hohe Präzision auf, was eine Zusammenführung von Antworten aus unterschiedlichen Quellen begünstigt.

4.2 Nutzerstudien

Die Kernevaluation erfolgte in drei Nutzerstudien - einer Pilotstudie, einer Hauptstudie und einer „Grand Challenge“. Die Motivation für mehrere kleine Studien gegenüber einer groß angelegten findet sich in mehreren Studien zur idealen Probandenanzahl, etwa in [DM90] oder einer Zusammenfassung von 16 Studien durch Nielsen und Landauer

[NL93], in der festgestellt wurde, dass das ideale Kosten-Nutzen-Verhältnis bereits mit 4 Teilnehmern erreicht wird und eine Sättigung bei etwa 16 Subjekten eintritt. In gewisser Weise tritt das Divide-and-Conquer-Prinzip von Softwarearchitektur und Benutzeroberfläche somit auch beim Studienentwurf in Erscheinung.

Die Pilotstudie folgt einem Within-Subjects-Design mit 12 Teilnehmern, die eine frühe Version des Ensembles mit Watson verglichen und dann zu ihrem allgemeinen Eindruck von QA-Systemen befragt wurden. Alle Alters- und Bildungsstufen außer „Bachelor at University of Applied Science“' wurden abgedeckt ebenso wie alle Sprachfertigkeitsstufen in Englisch. Das Interesse war auffällig hoch mit 812 gestellten Fragen, was 67.7 Fragen pro durchschnittlichem Nutzer entspricht. Während die kürzeste Bearbeitungszeit 24 Minuten betrug, verbrachten einige Nutzer freiwillig über 2 Stunden damit, mit dem System zu experimentieren. Das frühe Ensemble wurde durchgängig als vielversprechend (7) oder bereits nützlich (5) bewertet, wohingegen Watson konsistent als schneller und von 8 als bereits nützlich, von 3 Nutzern aber auch als noch viel Arbeit benötigend eingeschätzt wurde. Watson wurde dabei vor allem als nützlich für Erklärungen charakterisiert (5 vs. 1, sowohl für den Medizin- als auch Reisekorpus). Unerwartet wurde jedes der beiden Systeme von 5 Probanden als Unterstützung bei der Forschung in Erwägung gezogen, wohingegen niemand die Systeme als Spielzeug oder technische Revolution ansah. 8 beschrieben die Systeme als wertvolle Ergänzung zu Suchmaschinen und drei sahen sogar das Potenzial, Suchmaschinen zu ersetzen. Insgesamt sahen 11 Nutzer die Systeme als grundsätzlich hilfreich in der Lehre und 7 in der Forschung an.

Die Hauptstudie verglich YodaQA ohne Modifikationen mit dem vollen PalmQA Ensemble, basierte diesmal auf einem Between-Subjects-Design und wurde in einem lateinischen 2×2 Quadrat systematisch variiert, um auszugleichen, welches System zuerst benutzt wurde und damit als Vergleichskandidat fungierte. Die 16 Teilnehmer waren gleich auf die Geschlechter unterteilt. Auf einer 5-stufigen Likert-Skala bewerteten die Probanden das PalmQA Ensemble klar als schneller (med. 4.5, avg. 4.44) und berichteten nahezu übereinstimmend, dass es mehr korrekte Antworten liefere (5, 4.81). Ebenso wurden Antwortqualität (4, 4.31), Antwortdetails (4, 4.06) und die Nachrichtenmetapher (4, 4.19) positiv bewertet, die vordefinierten Antworten wurden sogar als exzellente Ergänzung eingeschätzt (5, 4.625).

Die finale Nutzerstudie wurde als „Grand Challenge“ an IBMs Jeopardy-Herausforderung angelehnt, in der 26 Probanden (13 männlich, 13 weiblich) bei 30 Faktenfragen, die ein nicht in das Projekt involvierter Kollege frei verfasst hat, gegen das System antraten. Der bestplatzierte Mensch schaffte es, 24 Fragen korrekt zu beantworten, was deutlich über dem Durchschnitt von 11 Fragen lag. Dennoch war das PalmQA Ensemble in der Lage, mit 25 Fragen knapp zu gewinnen und mehr als doppelt so viele Antworten wie der Durchschnitt der Probanden zu liefern, was statistisch hochgradig signifikant ist: Ein einseitiger t-Test (single sample) zum Signifikanzniveau 0.01 liefert einen p-Wert unter 0.00001.

5 Ergebnisse

Während das PalmQA Ensemble mit mehr als doppelt so vielen korrekten Antworten wie das beste offene Einzelsystem eine „übermenschliche“ Antwortrate aufweist mit durchschnittlicher Antwortzeit unter 10 Sekunden, was einer Minderung um fast einen Faktor von 3 entspricht, hat es auch klare Grenzen aufgezeigt. Es ist im Stande, etwa drei von vier Fragen richtig zu beantworten und dabei Quellen wie Wikipediaartikel zu liefern, was einen Startpunkt für weitere Recherchen in Forschung und Lehre bietet, verfügt aber über kein tiefes Verständnis, um ein Feld umreißen oder eigene Erklärungen formulieren zu können oder selbst sinnvolle Zusatzinformationen anzumerken, wo es hilfreich scheint. Aufgezeigte Erweiterungen wie das Abfangen vordefinierter Fragen können diese Unzulänglichkeiten kaschieren und somit die Akzeptanzrate steigern, sind aber nur temporäre Hilfslösungen. Auch die Nachrichtenmetapher verbessert das Basissystem, da sie ein Thema interaktiv erkundbar macht. Als Fazit scheint es für die Lehre ratsam, Lehrpersonal aktiv in das Antwortsystem einzubeziehen. Die größte Herausforderung im Hinblick auf die Zielsetzung dieser Arbeit bleibt jedoch die Forschung. Während momentane Antwortraten für Fragen zu Allgemeinwissen hinreichend sind, muss ein System für die Forschung vor allem domänenspezifisches Wissen zugänglich machen. Während Distributional Semantics, Keyphrase Extraction und Dokumentenkonvertierung erste Schritte in diese Richtung bieten, bleibt ein langer Weg, insbesondere, da zugrundeliegende Systeme wie YodaQA das Format von Title-Oriented-Documents oder die Verfügbarkeit von Ontologien annehmen, was die Domänenanpassung erschwert.

Die Eigenschaft, dass das PalmQA Ensemble sowohl Listen- als auch Einzelantworten formen kann, wurde bisher nicht ausgeschöpft. Dabei eröffnet es die Möglichkeit, auf die Zielgruppe zugeschnitten zu antworten. Nach der Erfahrung mit Watson-QA-APIs ist es für Forscher deutlich nützlicher, Textpassagen statt isolierter Antworten auszugeben und sie sind auch eher bereit, eine Liste von Antwortkandidaten durchzugehen.

Summa summarum bietet das System eine neuartige Plattform, um QA-Systeme in Forschung und Lehre nutzbar zu machen. Mit technischen Fortschritten im Cognitive Computing allgemein und in Dialogsystemen im Speziellen kann dieser Grundstock sukzessive erweitert werden, bis das Ziel vom intelligenten Assistenten für Fachwissen erreicht ist.

Literaturverzeichnis

[AFF15] Atapattu, Thushari, Katrina Falkner, and Nickolas Falkner: Artificial Intelligence in Education: 17th International Conference, AIED 2015, Madrid, Spain, In . C. Conati, N. Heffernan, A. Mitrovic, and F.M. Verdejo, eds. Pp. 13–22. Cham: Springer International Publishing. http://dx.doi.org/10.1007/978-3-319-19773-9_2, 2015.

[BR13] Biemann, C., Riedl, M.: Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity. Journal of Language Modelling 1(1): 55–95, 2013.

[BŠ15] Baudiˇs, Petr; Šedivy, Jan: Experimental IR Meets Multilinguality, Multimodality, and Interaction: 6th Int. Conference of the CLEF Association, CLEF'15, Toulouse, France, September 8-11, 2015, Proceedings. Springer International Publishing, Cham, Kapitel Modeling of the Question Answering Task in the YodaQA System, S. 222–228, 2015.

[DM90] Dalal, S. R; C. L. Mallows: Some Graphical Aids for Deciding When to Stop Testing Software. IEEE Journal on Selected Areas in Communications 8(2): 169–175, 1990.

[Fe09] Ferrucci, David, Adam Lally, Karin Verspoor, and Eric Nyberg: Unstructured Information Management Architecture (UIMA) Version 1.0, OASIS Standard, 2009.

[Fe10] Ferrucci, David A., Eric W. Brown, Jennifer Chu-Carroll, et al.: Building Watson: An Overview of the DeepQA Project. AI Magazine 31(3): 59–79, 2010.

[Gr61] Green, Bert F., Jr., Alice K. Wolf, Carol Chomsky, and Kenneth Laughery: Baseball: An Automatic Question-Answerer. Western Joint IRE-AIEE-ACM Computer Conference Pp. 219–224. IRE-AIEE-ACM '61 (Western). New York, NY, USA: ACM. http://doi.acm.org/10.1145/1460690.1460714, 1961.

[Gu09] Gurevych, Iryna, Delphine Bernhard, Kateryna Ignatova, and Cigdem Toprak: Educational Question Answering Based on Social Media Content. In Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling Pp. 133–140. Amsterdam: IOS Press. http://dl.acm.org/citation.cfm?id=1659450.1659474, 2009.

[Ku99] Kupiec, Julian M.: Natural Language Information Retrieval. In . Tomek Strzalkowski, ed. Pp. 311–332. Dordrecht: Springer Netherlands, 1999.

[NL93] Nielsen, Jakob, and Thomas K. Landauer: A Mathematical Model of the Finding of Usability Problems. In Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems Pp. 206–213. CHI '93. New York, NY, USA: ACM. http://doi.acm.org/10.1145/169059.169166, 1993.

[Pol16] Pollok, Falk: Leveraging UIMA-Based Question Answering Systems for e-Learning and Research. Master's Thesis, RWTH Aachen, Learning Technologies, 2016.

[Pr03] Prager, John M.; Chu-Carroll, Jennifer; Czuba, Krzysztof; Welty, Christopher A.; Ittycheriah, Abraham; Mahindru, Ruchi: IBM's PIQUANT in TREC2003. In (Voorhees, Ellen M.; Buckland, Lori P., Hrsg.): TREC. Jgg. Special Publication 500-255. National Institute of Standards and Technology (NIST), S. 283–292, 2003.

[Wa06] Wang, Chun-Chia, Jason C. Hung, Che-Yu Yang, and Hsuan-Pu Chang: A Question Answering System Approach for Collaborative Learning. In CSCWD 2006, Southeast University, Nanjing, China Pp. 1403–1407.

[We12] Wen, Dunwei, John Cuzzola, Lorna Brown, and Dr Kinshuk: Instructor-Aided Asynchronous Question Answering System for Online Education and Distance Learning. The International Review of Research in Open and Distributed Learning 13(5): 102–125, 2012.

[Wo73] Woods, W. A.: Progress in Natural Language Understanding: An Application to Lunar Geology. In: Proceedings of the June 4-8, 1973, National Computer Conference and Exposition. AFIPS '73, ACM, New York, NY, USA, S. 441–450, 1973.