Handbuch

Alexa Skill „Wismarer Nachrichten“

Von: Patrice Matz

E-Mail: p.matz@stud.hs-wismar.de

Modul: Grundlagen der Sprachtechnologien

Betreuende Professorin: Prof. Dr. Raab-Düsterhöft

Inhaltsverzeichnis

1 Idee

Dieser Skill soll es ermöglichen nicht nur die üblichen Nachrichten der Tagesschau u. ä. von Alexa vorlesen zu lassen, sondern von möglichst beliebigen Nachrichtenseiten.

Beim Lesen der aktuellen Nachrichten kann es jedem verziehen werden schlecht gelaunt zu sein. Nicht jeder möchte seinen Morgen mit negativen Nachrichten beginnen, daher wurde eine Funktion zum Bewerten des Sentiments eines Artikels hinzugefügt. Bevor ein Artikel vorgelesen wird kann das Sentiment ausgewertet werden. Somit hat der Nutzer die Wahl ob er einen negativen Beitrag hören möchte.

2 Funktionen

Dieser Alexa Skill besteht aus 5 Funktionen, zusätzlich zu den 5 Standard-Intents. Diese Sollen den Dialog Flow in Abbildung 1 ermöglichen

search_on	Setzt die Seite in der Session auf der gearbeitet werden soll.
news	Sucht nach Links zu Artikeln auf der Nachrichtenseite
search_for	Sucht nach Links zu Artikeln in der Suche
search_answer	Liest den ausgewählten Beitrag vor. Der Link zum Artikel wird der Session entnommen.
sentiment	Bewertet das Sentiment des gewählten Beitrags. Der Link zum Artikel wird der Session entnommen.

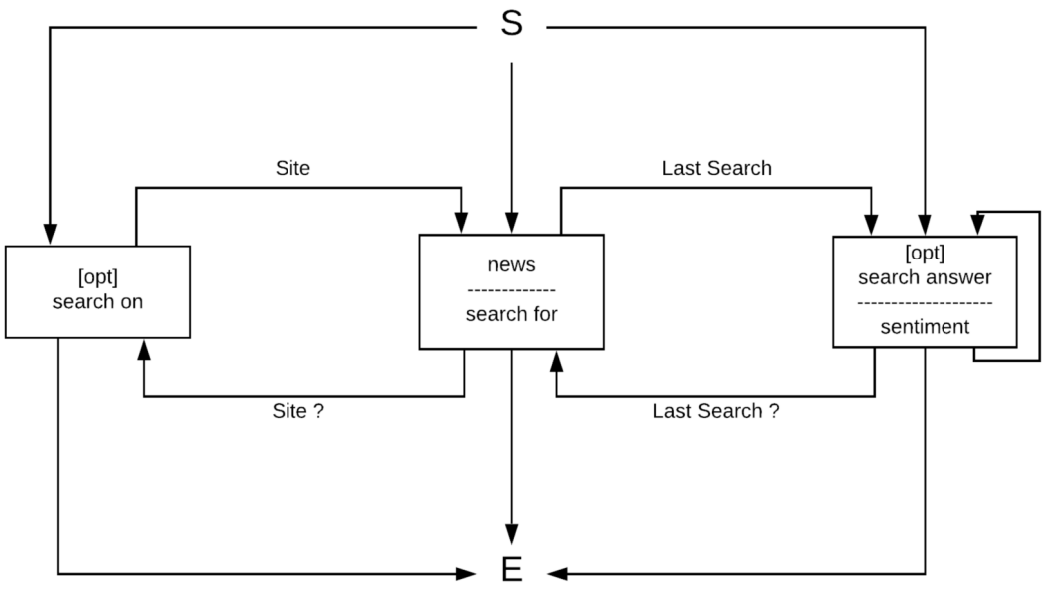

Abbildung 1: Dialog Flow

Der Nutzer kann zu jeder Zeit jeden Intent auslösen. Existieren die benötigten Informationen nicht in der Wissensbasis (Session) so wird der vorherige Intent aufgerufen, welcher diese bereitstellen würde.

2.1 Beispiel Dialog

Ein Dialog könnte wie folgt aussehen:

Invocation: Wie ist das Sentiment in der 1?

Ausgelöste Funktion: get_sentiment(number)

Response: Dieser Artikel existiert nicht. Wonach wollen Sie suchen?

Invocation: Gib mir die Nachrichten.

Ausgelöste Funktion: news(site)

Response: Auf welcher Seite wollen Sie hiernach Suchen?

Invocation: Auf Golem.

Ausgelöste Funktion: search_on(site) → news(site)

Response: [Die Überschriften der Nachrichten auf der entsprechenden Seite]

Invocation: Wie ist das Sentiment in der 1?

Ausgelöste Funktion: get_sentiment(number)

Response: Das Sentiment ist eher negativ / positiv.

Invocation: Danke.

Ausgelöste Funktion: StopIntent

Response: {}

Der Nutzer fragt das Sentiment im ersten aufgezählten Artikel ab. Da noch keine Suche nach Nachrichten oder einem Thema durchgeführt wurde existiert noch kein erster Artikel. Daher fordert get_sentiment() den Nutzer dazu auf zu sagen wonach er suchen möchte, die Antwort hierauf löst news() aus. An dieser Stelle hätte der Nutzer auch sagen können „Gib mir die Nachrichten auf golem“ da er dies nicht getan hat erfragt news() auf welcher Seite gesucht werden soll. Der Nutzer antwortet mit „golem“, woraufhin in der session nach der letzten aufgerufenen Funktion gesucht wird, da dies news() ist, wird news(site) mit der angegebenen Seite „golem“ aufgerufen. Jetzt kann der Nutzer sich entscheiden welchen Artikel er bewertet oder vorgelesen haben möchte. Die Session wurde anschließend mit „Danke“ geschlossen.

3 Implementierung

Jeder Alexa Skill besteht, wenn man so will, aus einem Frontend und einem Backend. Das Frontend stellt die model.json da, in ihr werden Invocations, Datentypen, usw. abgelegt. Das Backend ist der genutzte Endpoint. Der Endpoint kann in jeder beliebigen Sprache geschrieben werden, da er ein Webserver ist. In diesem Fall wurde Python und Flask gewählt. Flask-Ask ist eine Erweiterung für Flask welche das Handeling von Anfragen und Responses vereinfacht.

Jede Funktion beginnt mit dem Prüfen bzw. Beziehen der nötigen Inputparametern aus dem Slot oder der Session. Sollten diese nicht vorhanden sein wird der Nutzer dazu aufgefordert diese zu füllen.

Sind alle nötigen Informationen vorhanden wird ein Siteobject erstellt welches die benötigten Funktionen bereitstellt z. B. „Golem“ oder „Spiegel“.

3.1 Flask-Ask

Flask-Ask vereinfacht den Umgang mit den, von Amazon gesendeten, JSON Anfragen deutlich. Leider liegt die letzte Änderung ca. ein Jahr zurück. Dies kann daran liegen das seit dem keine Änderungen nötig waren, oder daran das der Entwickler das Projekt aufgegeben hat. Momentan liegen hierzu keine Informationen vor. Dadurch gestaltet sich die Installation schwieriger als üblich. (Die Installation ist nur nötig wenn nicht mit Docker gearbeitet wird)

```
python -m pip install pip==9.0.3
pip install flask-ask
pip install cryptography==2.1.4
pip install --upgrade pip
```

Code 1: Flask-Ask Installation

3.2 Klassen

Bisher wurden 2 Klassen implementiert. Jede Klasse steht für eine Website.

Es gibt eine Parent-Klasse welche die nötigen Funktionen enthält. Jede dieser Funktionen benötigt mindestens einen XPath Ausdruck, dieser muss in den erbenden Klassen angegeben werden.

Die XPath Ausdrücke können einfach mit den Browser Entwicklerwerkzeugen extrahiert werden.

4 Deployment

Für das Deployment gibt es zwei gleichwertige Möglichkeiten, die einfache und die ohne Docker.

Docker ist eine Container Technologie die es ermöglicht Software mit allen Dependencies auszuliefern. Zum Deployment nötig sind die Befehle im Folgenden:

```
docker build . -t gst
docker run -ti --rm -p 5001:5000 gst
```

Code 2: Deployment mit Docker

Die -p Flag bestimmt das Portforwarding, hier wird Port 5000 des Containers auf Port 5001 des Hosts geleitet. Port 5001 sollte durch den gewünschten Port ersetzt werden.

Der traditionelle Weg erfordert einen Webserver wie Nginx und Uswgi, sowie die manuelle Installation der Dependencies, welche in der requirements.txt aufgeführt sind und Flask-Ask. Es wird angenommen das dieses Know-How besteht, da dies den Inhaltlichen Rahmen an dieser Stelle sprengen würde.

Amazon fordert die Verwendung von Zertifikaten, es wird angenommen das ein Reverse Proxy eingesetzt wird um das Zertifikat Handeling zu übernehmen.

5 Erweiterung

Durch die Verwendung von einer Parent-Klasse von welcher geerbt wird ist die Erweiterung weiterer Seiten sehr einfach.

Schritt 1 ist relativ zeitaufwendig. Es muss eine neue Klasse erstellt werden (in siteobj.py), welche von „Site“ erbt, analog zu Spiegel oder Golem. Anschließen müssen alle Variablen und XPath Ausdrücke für diese Seite eingetragen werden.

Schritt 2: die get_site_obj() Funktion in Zeile 18 in app.py muss ergäntzt werden. Diese Funktion stellt die eigentlichen Objekte bereit mit welchen die Intents arbeiten. Hier muss ein „elif“ für die neue Seite eingefügt werden.

Schritt 3: die model.json enthält den type „Seite“ mit den Werten spiegel, welt, zeit und golem. Hier muss der Name der gewählten Seite eingetragen werden.

Schritt 4 ist das trainieren des Modells in der Developer Console.