

자연어처리를 기반으로 한 코로나 정보 제공 챗봇 시스템

송호연¹ · 곽찬우² · 이동원³ · 이윤수⁴

¹한성대학교 정보시스템공학과 · ²명지대학교 정보통신공학과

³수원대학교 컴퓨터학부 · ⁴서울여자대학교 콘텐츠 디자인학과

¹hoyeon94614@gmail.com / ²yhntfx777@naver.com

³ehddnjsgg@gmail.com / ⁴lightsoo0157@gmail.com

Informational Chatbot System about COVID-19 based on Natural Language Processing

Ho-yeon Song¹ · Chan-woo Gwak² · Dong-won Lee³ · Yun-soo Lee⁴

¹Dept, of Information System Engineering, Hansung University

²Dept, of Information and Communication Engineering, Myongji University

³Dept, of Computer Science, Suwon University

⁴Dept, of Contents Design, Seoul Woman University

요 약

본 논문에서는 코로나 바이러스로 인한 불편함을 겪는 사람들에게 코로나 관련 정보를 편리하게 제공하기 위해 인공지능 기반의 챗봇을 개발하였다. 인공지능 기술이 도입됨에 따라 챗봇이 여러 기관에서 고객 응대를 포함한 다양한 업무를 수행하고 있다는 점에 착안하여 IBM Watson Assistant를 활용한다. Watson Assistant는 사용자가 입력한 말을 자연어 처리를 통해 분석하여 문장을 생성한 후 사용자에게 전달한다. Intent와 Entity를 통해 질의의 행위와 주체를 입력한 후 높은 빈도가 예상되는 질문을 작성하고, Dialog를 통해 대화 흐름을 파악한다. 사용자는 Axure로 설계된 사용자 인터페이스를 통해 대화 전송, 정보 수신 등 동작을 취할 수 있으며, 각 언어에 맞는 SDK 라이브러리를 제공한다는 이점을 활용하여 Node.js로 화면에서 발생하는 액션과 데이터 전달을 처리한다.

1. 서론

최근 세계적으로 유행하는, 중국 우한에서 처음 발생한 코로나 바이러스로 불편함을 겪는 사람들이 한국에서도 늘어나고 있다. 본 논문에서는 많은 사람들이 알아야만 하는, 코로나 관련 정보를 쉽고 빠르게 얻을 수 있도록 IBM Watson에서 제공하는 기술로 대화형 인공지능 챗봇을 개발하였다. 대시보드로 사람들이 정보를 효과적으로 확인할 수 있도록 사용자 경험을 고려하여 UI를 디자인하였다. 또한 구현 과정에서 Node.js를 활용하여 챗봇이 서버에 접근할 수 있도록 하였다.

2. 시스템 기획 및 구성

본 논문에서는 챗봇을 통해 정보를 얻기 위해 기존의 검색을 하여 여러 사이트에서 별개의 정보들을

모으거나, 메뉴를 클릭하여 찾는 정형화된 이벤트에서 벗어나 챗봇의 목적인 인공지능 기반의 친근하고 인간형 에이전트 구현에 알맞게 사용자와 상호작용하는 대화형 시스템[1] 개발을 추구한다. Watson Assistant가 제공하는 자연어처리 기능을 통해 사용자의 질의를 이해하고 그에 따른 답변을 제공하는 것에 중점을 두었다.

시스템에 대해 설명하기에 앞서 인지 컴퓨팅과 Watson Assistant에 대하여 주목할 필요가 있다.

인지 컴퓨팅[2]이란 딥러닝(Deep Learning)방법론을 이용한 방식인데, 상세한 세부규칙들을 포함하는 프로그램을 작성하지 않아도 다양한 입력 데이터와 결과물의 정보를 제공하면 컴퓨터가 스스로 입력 데이터와 결과물을 연결하는 패턴 및 전략을 학습할 수 있도록 능력을 부여하는 분야이다.

Watson Assistant는 자연어 처리에서 딥 러닝

기술을 활용한 시스템으로 인간의 방대한 지식을 자연어 기반으로 받아들이며, 인간의 이해 수준까지 지식을 학습하고, 축적된 지식 Base를 활용하여 인간의 언어로 최적의 결과를 제공하는 인지컴퓨팅 서비스이다. 데이터 중심의 4세대 R&D 패러다임의 선두주자인 인지컴퓨팅 기술을 활용한 Watson Assistant를 통해 챗봇 시스템을 개발함으로써 전통적 정보 수집 방식에서 벗어나 보다 효율적으로 사용자들이 정보를 수집할 수 있도록 하는 것을 본 챗봇 시스템의 목적으로 한다.

개발 전 개발범위를 계획하고, 챗봇 시스템이 수행해야 하는 기술을 정의하기 위해 요구사항을 작성한다. 요구사항은 중요도에 따라 우선순위를 하여 자원 문제, 요구사항 추가 등이 발생할 때 요구사항 선별 및 구현순서를 원활히 조정하고[3], 이용하는 주체에 따라 시스템, 사용자로 나누어 작성하였다. 작성된 요구사항을 토대로 시나리오를 설계하는데, 시나리오는 후에 제작할 Dialog의 완성도를 검증할 때 이용한다.



<그림 1> 챗봇 시스템 시나리오

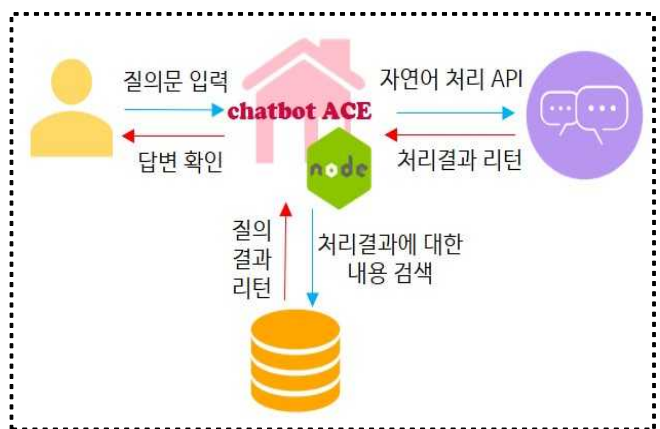
<그림 1>에서는 요구사항을 토대로 챗봇 시스템이 제공하는 기능과 각 기능 수행 시 나타나거나 사용자가 취할 수 있는 다양한 결과를 예측하는 시나리오를 작성하였다.

3. 챗봇 설계 및 동작

Watson Assistant는 사용자가 대화창에 입력한 질의에서 기존의 학습을 바탕으로 Intent와 Entity를 구별하고, 연관성이 있는 Dialog를 생성하는 자연어 처리 과정을 거친 후 문장을 생성하여 Node.js에게 전달한다. Node.js는 UI와 Watson Assistant 사이에서 각각 요청과 응답을 전달하는 역할을 맡는다.

Watson Assistant의 자연어 처리 후 결과를 리턴받은 Node.js는 연동되어있는 데이터베이스에서 처리결과와 동일한 답변이 있는지 확인한다. 일치하는 데이터가 존재할 경우, 데이터베이스는 Node.js에게 질의결과를 리턴하고 최종적으로 사용자가 답변을 받아볼 수 있다.

사용자는 입력한 질의의 종류에 따라 다양한 형태의 응답을 받아볼 수 있다. 간단한 질의는 intent로 학습되어 저장된 문장을 응답으로 받아볼 수 있고, 세부적인 정보 제공이 필요하다고 판단되는 질의는 웹에서 제공하는 정보 포함 여부에 따라 외부링크 또는 해당 메뉴로 연결되는 링크로 받아볼 수 있다.



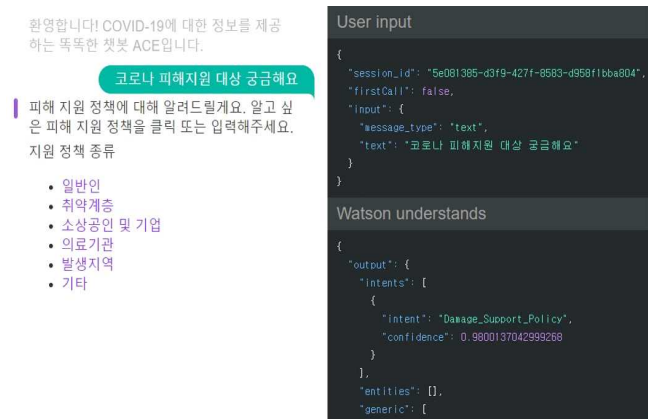
<그림 2> 챗봇 동작 원리

사용자가 가장 많이 궁금해하는 예방 수칙, 확진자 동선, 관련 뉴스와 같은 정보를 제공할 수 있도록 Dialog를 구성하였다.

<그림 3>에서는 사용자 요구사항을 바탕으로 웹에서 제공할 기능을 파악하여 Intent name에 담고, 인공지능의 학습을 위해 사용자가 질문할 만한 질의예제를 입력하고 등록한다. 학습 과정을 거치고, Intent의 설계가 끝나면 Dialog에서 작성한 Intent를 바탕으로 챗봇이 사용자에게 제공할 답변을 예상하여 자연스러운 대화의 흐름을 구성한다. [4, 5]



<그림 3> Intent와 Dialog 구성

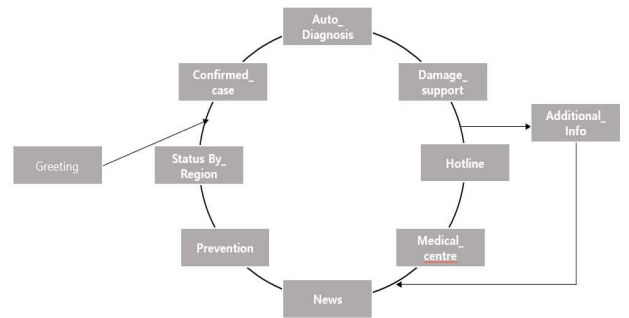


<그림 4> localhost상 챗봇 테스트

Dialog의 구성을 마치면 <그림 4>처럼 localhost 상에 챗봇을 띄우는 작업을 진행한다. 해당 Assistant의 ID와 API를 이용하여 환경설정을 구성한다. 이 과정이 끝나면 부여받은 포트 번호를 통해 Web application이 실행된다. 채팅창의 우측에는 Watson API의 호출 방식을 확인할 수 있는데 주로 JSON 형태로 처리되어 입력값은 input으로, 챗봇이 답변하는 내용은 output으로 처리되어 나타난다.

‘코로나 피해 지원대상’ 궁금해요.’라는 질의를 입력하였더니 Watson Assistant가 해당 문장에서 사용자의 Intent, 즉 피해로 인한 정책과 대상 관련 질문이라는 사용자의 의도를 파악하고 ‘지원’, ‘정책’이라는 Entity를 추출한다. Dialog를 설계할 때 Entity와 Intent의 유형에 따라 Condition항목을 통해 조건에 따라 사용자에게 보여질 문장을 다르게 설정할 수 있다. 또한, Response Type 항목을 여러 개 추가하여 사용자에게 폭넓은 선택권을 제공하는 것이 가능하다. <그림 4>에서는 반환된 답변을 확인했더니 사용자의 신분 또는 소속에 따라 분류된 지원 정

책 목록을 제공하는 것을 확인할 수 있다. 그리고 전달받은 질의 내에서 Entity와 Intent를 인지하고 해당 키워드가 포함된 정보를 외부링크 또는 시각적인 효과를 위해 이미지, 영상 등 미디어 파일, 웹 메뉴로 연결되는 링크로 받아 볼 수 있다. 또한, 해당 시스템은 크롬, 파이어 폭스와 같은 다양한 브라우저에서 호환될 수 있다.



<그림 5> 챗봇 dialog 흐름도

각 Entity는 <그림 5>와 같은 구조로 연결되어 있다. 초기 인사화면 이후 입력하는 정보에 따라 원안의 해당하는 Entity에 접근하여 답변을 출력하고, 추가정보 Entity를 거친 이후에 또 다른 추가적 정보에 접근할 수 있다. 이러한 흐름을 통하여 원하는 정보를 대화 형태로 얻을 수 있다.



<그림 6> 구현된 웹 UI

Dialog와 시나리오를 포함한 설계도를 바탕으로 구현된 웹의 UI에 사용자가 대화창에 질의를 입력하거나, 메뉴를 통해 직접 해당 페이지로 이동할 수 있도록 두 가지 방식의 구성을 하였다.

4. 결론

본 논문에서 구현한 자연어처리를 기반으로 한 코로나 정보 제공 챗봇 시스템은 IBM사의 Watson Assistant의 와 Node.js를 이용하여 사용자가 원하는 정보를 웹을 통해 간편하고 빠르게 제공받을 수 있도록 한다. 사용자는 단순히 웹 상에서 정보를 검색하여 수집하는 것이 아니라, 실시간으로 이루어지는 챗봇과의 대화를 통해 실제 전문가와 소통하고 있다는 느낌을 받을 것이다. 이러한 상호작용을 통해 친밀감을 높인다면 챗봇을 이용하려는 사용자의 증가를 기대해 볼 수 있다.

또한, 챗봇을 이용하는 다양한 사용자들의 언어 습관을 고려하여 많은 양의 학습을 반복적으로 시킨다면 인공지능의 문장 인지 능력을 향상시킬 수 있을 것이며 본 논문에 기재된 기술을 응용하여 유사 질병 대책에 관한 전반적인 정보를 얻을 수 있는 플랫폼으로 확장하는 방향으로 발전시킬 수 있을 것이다.

Acknowledgement

이 논문은 ICT멘토링 프로그램에서 지원하는 한이음 프로젝트와 (주)글로벌링크스의 정영해 대표님과 함께 수행되었음.

‘본 논문은 과학기술정보통신부 정보통신창의인재 양성사업의 지원을 통해 수행한 ICT멘토링 프로젝트의 결과물입니다.’

참고문헌

- [1] Seongeun Kim, Mincheol Shin, Juyoung Kang, “Introduction of Chatbot Technology and Case Analysis”, KICS Information & Communication Magazine - Open Lecture Series, Vol. 35, No. 2, pp. 21 - 28, Nov. 2018
- [2] Seongho Lee, “Cognitive Computing”, Science and Technology Policy Institute, SCIENCE & TECHNOLOGY POLICY 26(5), 2016.5, 16-23(8 pages)
- [3] Jaeseok Seong, Dongsu Kang, Cheeyang Song,

Dookwon Baik, “A Design of Process for Requirement Prioritization”, The Korean Institute of Information Scientists and Engineers, Vol. 35, No. 1, pp. 13 - 18, June. 2008

- [4] Bo-Keom Kim, Tae-Wook Kim, Ho-Yeon Kim, Gyung-Ho Hwang, Kyo-Kun Kim, Ji-Yeon Kang, Chang-Dae Cho, “Development of AI Online shopping mall Chatbot based on IBM Watson Assistant”, in Proceedings of Symposium of the Korean Institute of communications and Information Sciences, Korea Institute Of Communication Sciences, pp. 628-629, Nov. 2018
- [5] IBM Cloud. Getting started with Watson : <https://cloud.ibm.com/docs/assistant?topic=assistant-getting-started#getting-started>