

다양한 문장 생성을 위한 Learning Classifier System 기반 대화형 에이전트

윤은경⁰ 조성배
연세대학교 컴퓨터과학과
ekfree@candy.yonsei.ac.kr⁰, sbcho@cs.yonsei.ac.kr

A Conversational Agent based on Learning Classifier System for Generating Various Types of Sentences

Eun-Kyung Yun⁰ Sung-Bae Cho
Dept. of Computer Science, Yonsei University

요 약

인터넷 이용자 수가 증가함에 따라 각 사이트에서는 수많은 정보들 중에서 사용자가 원하는 정보를 효과적으로 제공해야 할 필요성이 증가하고 있다. 이에 따라 사용자와 자연어로 정보를 주고 받으며 사이트의 가상 대리자 역할을 수행할 수 있는 대화형 에이전트에 관한 연구가 활발히 진행되고 있다. 하지만 대부분의 에이전트들이 단순한 패턴 매칭 기법을 사용하기 때문에 만족스러운 답변을 제공해주지 못하고 있다. 본 논문에서는 사이트에 대한 사용자의 친밀도를 높일 수 있도록 하기 위해, 자연어로 제공되는 사용자 질의에 대해 그 의도를 파악한 후 이에 대한 다양한 답변을 수행하는 대화형 에이전트를 제안한다. 이 때 대화의 지능성과 다양성을 보장하기 위하여 GA를 이용한 learning classifier system을 도입하여 주어진 데이터베이스로부터 새로운 문장을 만들어 다양한 답변을 할 수 있도록 한다. 연구실을 소개하는 간단한 문제에 적용하여, 그 가능성 및 개선점을 파악한다.

1. 서 론

인터넷 사이트의 정보량이 증가함에 따라 사용자에게 필요한 정보를 적절히 제공해 줄 필요성이 증가하고 있다. 이러한 목적을 위해 대부분의 사이트에서 키워드에 기반한 단순한 검색기법을 주로 사용하는데, 이 방식은 사용자의 의도를 제대로 파악하기가 어려울 뿐 아니라, 사용자가 자연스럽게 정보를 검색할 수 없는 문제가 있다. 또한 사용자의 사이트에 대한 친밀감도 감소시킬 수 있다. 이러한 단점을 극복하기 위해서 자연어에 기반한 대화를 통하여 사용자의 의도를 명확히 함으로써 정확하고 빠른 정보를 제공해 줌과 동시에 자연스럽게 편안한 대화를 할 수 있는 대화형 에이전트가 필요하다.

2. 관련 연구

최초의 대화형 에이전트라 할 수 있는 ELIZA는 자연어처리 연구를 위해 고안되었으며 단순한 패턴 매칭 기법을 사용하였다[1]. ALICE는 AIML이라는 언어를 사용하여 질문-대답 패턴을 기술하고 사용자의 질문에 해당하는 대답을 찾아 사용자와 대화한다[2]. 그러나 이 시스템은 질문을 답변에 매칭시킬 때 키워드에 기반하여 단순한 순차적 패턴 매칭을 사용하기 때문에 사용자의 의도를 반영한 답변을 정확히 제시하지 못할 수 있는 단점이 있다. 또한 답변 데이터베이스 구축시 많은 노력이 들어가야 한다. 이 외에도 Nativeminds사의 Nicole, Artificial Life사의 SmartBot, Virtual Personalities사의 Verbot 등이 현재 상용화되어 있다.

대부분의 대화형 에이전트들이 미리 정해져 있는 문장으로 말하기 때문에 사용자는 시간이 지남에 따라 쉽게 지루함을 느낄 수 있으며, 친밀감도 떨어뜨리게 된다. 이 연구에서는 Learning Classifier System(LCS)[3,4]의 환경 적응성을 이용하여 다양한 문장의 대화 능력을 갖춘 대화형 에이전트를 구현하고자 한다.

3. LCS

LCS[3,5]는 1978년 Holland와 Reitman에 의해 처음 소개된 기계학습 시스템으로, classifier라 불리는 규칙들을 GA를 이용해 학습시키는 시스템이다. 이는 유동적인 환경 요인을 가진 시스템을 구축하는 데에 적합하다고 할 수 있다[5]. 대략적인 구조는 그림 1과 같다.

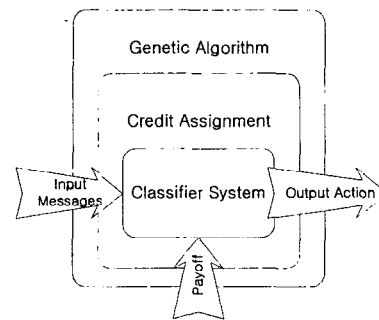


그림 1. LCS의 구조

Classifier system은 여러 개의 classifier, 즉 규칙들로 구성된다. 하나의 classifier는 조건부와 실행부로 나눌 수 있는데, 조건부는 {0,1,#} 세 종류의 비트로 이루어진 스트링이고 실행부는 {0,1}로 이루어진 스트링이다. #은 0 또는 1과 모두 매칭될 수 있다.

LCS는 입력 인터페이스를 통해 외부 메시지를 입력 받고 출력 인터페이스를 통해 시스템이 결정한 결과를 외부 환경으로 내보낸다. 환경은 출력으로 나온 결과에 대해 적절한 보상을 함

으로써 시스템에 피드백을 준다.

Classifier의 경쟁에는 strength라는 값이 사용되는데 strength가 높을수록 경쟁에서 이길 확률, 즉 선택될 확률이 크다. 또한 각 classifier는 specificity라는 값을 가지는데 이는 classifier의 조건부에 #이 아닌 비트 수를 의미한다.

LCS는 그림1과 같이, 크게 세 부분으로 구성된다.

• **Classifier System** : 외부 환경으로부터 입력된 메시지와 모든 classifier의 조건부를 비교하여 매칭되는 것을 찾는다. 비교는 비트-대-비트로 이루어진다.

• **Credit Assignment System** : 매칭된 classifier들은 경쟁을 통해 하나의 승자만이 선택되어 외부로 보내진다. 이 때 bid라는 값을 통해 각 classifier의 가치를 평가하는데 bid는 각 classifier의 strength와 specificity를 이용하여 구한다.

$$bid = c * specificity * strength$$

(c : 1보다 작은 양수)

외부로 보내진 액션에 대해 환경으로부터 피드백이 들어오면 승자 classifier의 strength가 조정된다.

• **Rule Discovery System** : Rule discovery system에서 사용되는 GA(genetic algorithm)[4]는 여러 번의 반복이 있는 후한 세대가 될 때 적용되는 것이 일반적이지만, 대화형 에이전트에서는 새로운 문장이 빨리 만들어져야 함을 감안해서 입력 문장이 들어올 때마다 GA를 적용한다.

매칭된 classifier가 있다면 승자 classifier와 리스트에 있는 나머지 문장 중 하나를 임의로 선택하여 교차를 수행, 생성된 새로운 classifier들은 classifier list에 추가하고, 만약 매칭된 classifier가 없다면, 임의로 하나의 classifier를 생성해서 추가한 후, 나머지 classifier들에 돌연변이를 취한다.

4. LCS기반 대화형 에이전트

이 논문에서 구현한 대화형 에이전트 시스템의 전체적인 구조는 그림 2와 같다. 사용자의 자연어 질의에 대해 전처리 모듈에서 철자 오류나 동의어 등을 처리한 후, 문형 분류 과정을 거쳐 사용자의 문형(Dialogue Act)[6]을 분류한다[7]. 분류된 문형정보와 사용자의 원래 질의를 바탕으로 해당하는 질문에 가장 적합한 주제를 선택하여 그 주제에 대한 classifier list에서 매칭하는 답변을 사용자에게 제시한다.

이 시스템에서 중심이 되는 기술을 보면 다음과 같다. 먼저 사용자가 입력한 문장에 대해 문형을 분류한다. 분류된 문형은 사용자의 대략적인 질의 의도 파악이 가능하고, 답변을 매칭할 때 사용자의 질의 의도에 적합한 답변을 매칭할 수 있게 한다. 그리고 사용자 질의에 대하여 미리 구축된 답변 데이터베이스로부터 적절한 답변을 매칭하여 이를 사용자에게 제시해 주어야 한다. 이 때 고정된 답변 문장이 아닌 유동성 있는 문장을 답변으로 내보내기 위해 LCS를 도입한다.

4.1 문형 분류

표 1과 같이 문형을 나누어 각 문형을 정의하는 일련의 키워드 리스트와 이들의 순서 정보를 이용하여 오토마타를 구성함으로써 분류를 수행한다.

표 1. 문형 분류 유형

질의 대분류	문형
1차 질문형	Can, Who, Method, Location, Reason, Should, Time, Description, etc
2차 질문형	Compare, Confirm, Cost, Direction, etc
1차 평서문형	Message, Act, Is, Have, Want, Fact, etc
2차 평서문형	Cause, Feeling, Time, Conditional, etc

하지만 각 문형별 오토마타를 독립적으로 적용하면 사용자 질의의 수는 많은데 각 문형에 대한 키워드 리스트는 제한되어 있기 때문에 하나의 질의가 여러 개의 문형으로 결정되어질 수 있다. 이를 해결하기 위해 포섭구조(subsumption architecture)[8]를 이용하여 각 분류 간에 서로 활성화, 억제화를 함으로써 관계 조정을 할 수 있도록 한다.

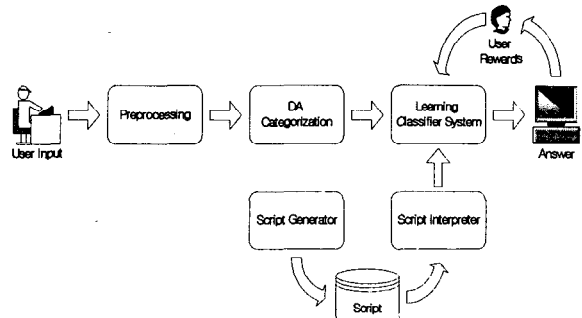


그림 2. 시스템의 전체적인 구조

4.2 답변 매칭과 LCS

에이전트는 사용자 질의에 대해 문형 분류 정보와 추출된 키워드 정보를 가지고 답변 문장 정보가 들어 있는 답변데이터 베이스에 접근한다. 답변 데이터베이스는 기본적으로 질문(조건)-대답의 형태를 가진 classifier system으로 이루어져 있으며 질문-대답의 기본 구성은 토픽을 기본단위로 하여 표현된다.

사용자 질의를 분석하여 얻은 문형정보와 문형별 키워드 정보, 사용자 질의를 가지고 토픽에 기술된 조건과의 매칭 점수를 계산한다. 이 때 조건부에 기술되어 있는 요소에 따라서 점수가 부여된다. 최고 점수를 가진 토픽이 선택되면 에이전트는 LCS를 구성하여 해당 토픽에 대한 classifier list 파일을 해석한다. 이 파일에는 조건부에 대한 정보와 답변 문장을 구성할 단어들, 그리고 초기에 생성된 classifier list가 들어 있다.

하나의 classifier를 구성하고 있는 조건부와 실행부는 다음의 표 2와 3과 같다. 조건부에는 해당 토픽과 관련된 문형 정보들에 대한 비트와 실행부에 해당하는 답변 문장의 구조를 선택할 수 있는 두 비트로 구성된다. 즉, 조건부의 마지막 두 비트는 답변 문장이 단순히 키워드만 포함하고 있는 형태나 주어, 서술어가 완벽한 문장 형태, 그 이외의 수식어 부분이 추가된 형태 등을 나타내는 역할을 한다. 현재의 시스템에서는 구현이 되어 있지 않지만 차후에 사용자 모델링 측면에서 에이전트를 개선하고자 할 때에 유용한 정보가 될 것이다. 실행부는 답변 문장을 구성하는 부분, 즉 주어, 키워드, 서술어 등으로 나누는데 토픽에 따라 주어와 서술어의 내용이 달라질 수 있다. 표4는 실제 classifier list의 예인데, 맨 오른쪽 열은 각 classifier의 초기 strength값이다.

표 2. Classifier의 조건부 구성 방식

조건부(n-bit)					
문형1	문형2	문형n-2	문형n-1	문형n

표 3. Classifier의 액션부 구성 방식

실행부(m-bit)		
주어부 a비트	키워드부 b비트	서술어부 c비트

사용자의 질의에 알맞은 토픽이 선택되고 해당 classifier list 파일의 해석이 끝나면, 에이전트는 질의를 분석한 정보를 LCS의 입력 메시지로 보고, 매칭되는 classifier를 찾는다. 매칭된 classifier가 존재하면 해당 답변을 내보내고, 그렇지 않으면 임의로 새로운 답변을 만들어 내보내고 이를 classifier list에 추가한다. 동시에 GA를 적용하여 교차와 돌연변이 연산으로 새로운 classifier를 생성하여 추가한다. 학습을 위한 적합도 평가는 strength값을 이용하는데, 사용자가 에이전트의 답변 문장에 대해 -2점부터 +2점까지의 점수를 피드백으로 주도록 하여, 이를 strength에 적용한다.

교차 연산자 적용의 경우, 선택된 두 개의 classifier에 대해 무작위로 교차 지점을 선택하게 되면 문장 구성이 망가질 위험이 있기 때문에 주어부와 키워드부, 서술부 내부에서는 교차가 일어나지 않도록 제한을 가하였다.

표 4. Classifier List의 예

1#####1011110	10
101#####0010101	10
#####11:0110101	10
#1###1###:0011101	10
#101####:1011010	10

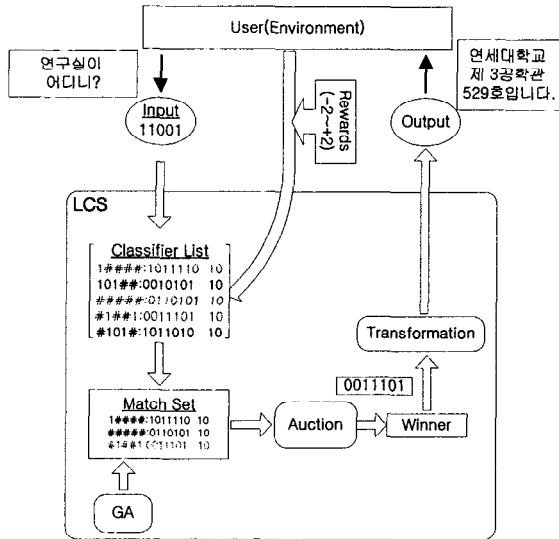


그림 3. LCS의 구조

5. 실험 결과

모든 classifier list의 초기 개체 수는 5개로 동일하게 했으며, 각 classifier는 수작업으로 초기화했다. GA 연산자 적용시 제한을 가하기 위해 주어부와 서술어부의 위치를 미리 데이터베이스화했다.

표 5는 연구실 위치와 관련한 일련의 대화를 기록한 것이다. 첫번째 대화는 같은 질문에 대하여 새로운 답변 문장이 생성되었음을 볼 수 있다. (괄호 안은 classifier 내부 표현이다.) 두 번째 대화는 사용자의 질문에 매칭되는 규칙이 없을 경우이다. 이때에는 에이전트가 임의로 규칙을 만들어 답변을 내보내는데, 사용자가 답변에 만족하는 정도에 따라 점수를 주게 된다. 여기서 서술부가 없는 답변 문장이 수행되었기 때문에 -2점의 페널티를 주었다. 그리고 나서 다시 같은 질문을 하였더니 완전한 문장이 답변으로 나오음을 볼 수 있었다.

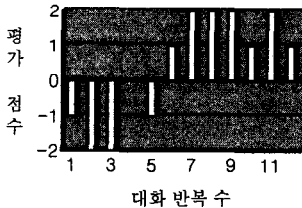


그림 4. 답변 문장에 대한 사용자 평가

실제로 여러 번의 대화를 반복한 후, 생성된 classifier를 분석해 본 결과 다양한 답변 문장들이 생성됨을 확인할 수 있었다. 또한 그림 4에서 보는 바와 같이 여러 번의 반복 학습 후에

표 5. 대화의 예

[대화1]
User : 소프트 컴퓨팅 연구실이 어디인가요? (1101100)
Agent : 저희 연구실은 연세대학교 제 3공학관 5층에 있습니다. (1011010)
User : 소프트 컴퓨팅 연구실이 어디인가요? (1101100)
Agent : 저희 연구실은 연세대학교 제 3공학관 5층 529호입니다. (1011101)
[대화2]
User : 연구실이 어디에 있는지 말해주세요. (0011001)
Agent : 저희 연구실은 서울시 서대문구 신촌동 134번지 529호 (1100100)
User : 연구실이 어디에 있는지 말해주세요. (0011001)
Agent : 저희 연구실은 서울시 서대문구 신촌동 134번지 5층 529호에 있습니다. (1101110)

에이전트는 사용자로부터 높은 점수를 받은 완전한 문장으로만 답하였다.

6. 결론

기존의 에이전트들은 정해진 문장을 그대로 출력으로 내보내기 때문에 비록 답변 문장들 중 임의로 선택되더라도 대화의 의의성이나 다양성에 있어서는 다소 부족한 면이 있었다. LCS를 결합한 이 시스템은 규칙 기반으로 되어 있는 스크립트를 classifier system으로 바꾸어 학습이 가능하도록 함으로써 새로운 문장을 생성하여 대화의 다양성과 유연성을 더 높였다고 할 수 있겠다. 하지만 이 시스템에서는 가지고 있는 단어들을 의미 단위로 나누어 새로운 문장을 구성한 것이기 때문에, 같은 질문에 대해 새로운 답변 문장이 몇 개 더 생성되기는 했지만, 에이전트가 문장 구조나 의미에 대한 지식을 충분히 갖추었다고 볼 수는 없다[9]. 따라서 문장을 구성하는 문법 구조 연구를 통해 에이전트 스스로 올바른 문장을 만들어 나갈 수 있는 능력을 갖도록 해야 하겠다.

7. 참고 문헌

- [1] J. Weizenbaum, "ELIZA - A Computer Program for the Study of Natural Language Communication between Man and Machine," *Communications of the ACM*, vol. 9, no. 1, pp. 36-45, 1965.
- [2] <http://alicebot.org>
- [3] L. B. Booker, D. E. Goldberg, and J. H. Holland, "Classifier Systems and Genetic Algorithms," *Artificial Intelligence*, Vol. 40, pp. 235-282, 1989.
- [4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley: Reading, MA, 1989.
- [5] J. H. Holland, "Using Classifier Systems to Study Adaptive Nonlinear Networks," *Lectures in the Science of Complexity*, SFI Studies in the Science of Complexity, D. Stein, (ed), Addison Wesley, 1989.
- [6] M. G. Core, and J. F. Allen, "Coding Dialogs with the DAMSL Annotation Scheme," *Working Notes of the AAAI Fall Symposium on Communicative Action in Humans and Machines*, pp. 28-35, 1997.
- [7] S.-I. Lee and S.-B. Cho, "A Conversational Agent Based on Structured Pattern Matching," *Proc. Korea Information Science Society (B)*, vol. 28, no. 1, pp. 409-411, Suwon, April 2001.
- [8] R. A. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation*, pp. 14-23, 1986.
- [9] R. De Mori, "Recognizing and Using Knowledge Structures in Dialog Systems," *Proceedings of IEEE-ASRU99(invited paper)*, 1999.