COBATO. Un chatbot orientado a asistir al pequeño comercio

COBATO. A chatbot aimed at assisting small retailers

Clara Díaz-Ruíz[1], Fernando Martínez-Santiago[1], Arturo Montejo-Ráez[1], Maria Teresa Martín-Valdivia[1], L. Alfonso Ureña-López[1], Manuel Carlos Diaz-Galiano[1], Miguel Angel Garcia-Cumbreras[1], Manuel García-Vega[1], Flor Miriam Plaza-del-Arco[1], Salud María Jiménez-Zafra[1] and María Dolores Molina-González[1]

[1]*Grupo SINAI, Departamento de Informática, Universidad de Jaén, Universidad de Jaén, Campus Las Lagunillas, 23071, Jaen, Spain*

Resumen

Se presenta COBATO, un chatbot cuyo dominio es el pequeño comercio y que tiene WhatsApp como canal de comunicación. La finalidad de COBATO es asistir al comercial en aquellas necesidades de información que plantean los clientes y que usualmente se resuelven vía telefónica o algún servicio de mensajería. Así, el asistente virtual facilita al cliente información de productos, horarios, datos de contacto, además de anotar pedidos. En el ámbito del Procesamiento del Lenguaje Natural, se aporta un modelo de datos basado en un grafo de conocimiento que aglutina toda la información que el chatbot requiere del dominio de la aplicación. Una segunda aportación es una representación formal basada en marcos gramaticales del lenguaje que el chatbot conoce. Estos son utilizados para el análisis semántico, así como para generar ejemplos de respuestas de usuario con las que entrenar el modelo de lenguaje usado en el flujo de compresión del lenguaje del chatbot.

COBATO is a chatbot intended by small commerce domain using WhatsApp as a communication channel. The purpose of COBATO is to assist the salesperson in order to provide information that is usually solved via telephone or a messaging service. Thus, the virtual assistant provides the customer with information on products, opening hours, contact details, as well as taking orders. In the field of Natural Language Processing, a data model based on a knowledge graph is proposed, which brings together all the information that the chatbot requires from the application domain. Additionally, a formal representation based on grammatical frameworks of the language that the chatbot knows is obtained. Subsequently, these are used for the semantic analysis of the user response. The fine-tuning of probabilistic language models is achieved by means of examples generated with the grammar.

Keywords

asistente virtual, chatbot, GF, modelos del lenguaje, PLN

1. Introducción

Con frecuencia, el limitado aforo del pequeño comercio conlleva largas esperas y colas para realizar compras domésticas cotidianas. En el contexto del pequeño comercio, el comercio de barrio, un modo de mitigar estas colas es realizar pedidos bien por teléfono o enviando un simple mensaje por WhatsApp, de modo que el cliente se acerca a por el pedido cuando este está confeccionado. Se propone el desarrollo de un chatbot, denominado COBATO, con el objetivo de apoyar al comercio en tareas propias de la atención al cliente: facilitar información del comercio y de productos así como la elaboración de encargos. COBATO está diseñado como una suerte de intermediario entre el cliente y el dependiente de modo que los tres actores comparten un mismo canal, WhatsApp en este caso. Ya en el ámbito específicamente del Procesamiento del Lenguaje Natural, o PLN, las principales aportaciones de COBATO se resumen en los siguientes puntos:

- Uso de marcos gramaticales (GF, *Grammatical Frameworks*)([R]. Concretamente, los GF proveen de un analizador semántico especializado al dominio de la aplicación. Sin

SEPLN-PD 2022. Annual Conference of the Spanish Association for Natural Language Processing 2022: Projects and Demonstrations, September 21-23, 2022, A Coruña, Spain

cdr00008@red.ujaen.es (C. Díaz-Ruíz); dofer@.ujaen.es (F. Martínez-Santiago); amontejo@.ujaen.es (A. Montejo-Ráez); maite@.ujaen.es (M. T. Martín-Valdivia); laurena@.ujaen.es (L. A. Ureña-López); mcdiaz@.ujaen.es (M. C. Diaz-Galiano); magc@.ujaen.es (M. A. Garcia-Cumbreras); mgarcia@ujaen.es (M. García-Vega); fmplaza@ujaen.es (F. M. Plaza-del-Arco); sjzafra@ujaen.es (S. M. Jiménez-Zafra); mmolina@ujaen.es (M. D. Molina-González)

embargo, este analizador, si bien es muy preciso, presenta una cobertura limitada, por lo que es necesario apoyarse en modelos del lenguaje como *word embeddings* o RoBERTa. A estos, la gramática implementada en GF provee de ejemplos de frases de usuario que son generados mediante un proceso denominado "linearización". Estos ejemplos están etiquetados con la acción y las entidades que allí se encuentran, y son utilizados para el ajuste o *fine-tuning* del modelo del lenguaje.
- Grafos de Conocimiento (KG, *KnowledgeGraphs*)([H], como modelo único para la representación del conocimiento, y que encapsula tanto el modelo de datos del dominio de la aplicación como el conocimiento lingüístico, el cual será interpretado por los GF, previa traducción automática de un modelo de datos a otro.

2. Solución propuesta

En la línea de ([F], la energía ([F] y ([C], se propone un grafo de conocimiento para enlazar toda la información, como un único formalismo que representa (i) la base de datos, donde se codifica la información estructurada que se desea hacer pública, y (ii) conocimiento lingüístico, que se representa como un conjunto de entidades y conceptos (nodos) y relaciones entre ellos (arcos). Este conocimiento lingüístico es posteriormete trasladado a GF, los cuales proveen de un compilador capaz de realizar análisis semánticos del texto, a la par que generar expresiones que son plausibles conforme la gramática codificada. Como se indicó en la introducción, estas facilidades son aprovechadas tanto para obtener una interpretación muy precisa de la respuesta de usuario como para el ajuste fino del modelo de lenguaje que se requiere en el flujo PLN del chatbot. A continuación se detallan las tecnologías que forman parte de la arquitectura de COBATO (ver Figura 1). COBATO requiere de diversas tecnologías, que se agrupan en servicios de back-end, servicios de front-end, así como ciertos recursos externos utilizados para dotar de conocimiento lingüístico a los servicios de back-end.

- Servicios de front-end
  - *React*. Front-end Web para la gestión de la página web que usará el comercio : perfil de negocio, horarios, productos, disponibilidad y precios.
  - *Venom*. Front-end WhatsApp, pasarela entre el chatbot y los dos perfiles de usuario de este, cliente y comercio.

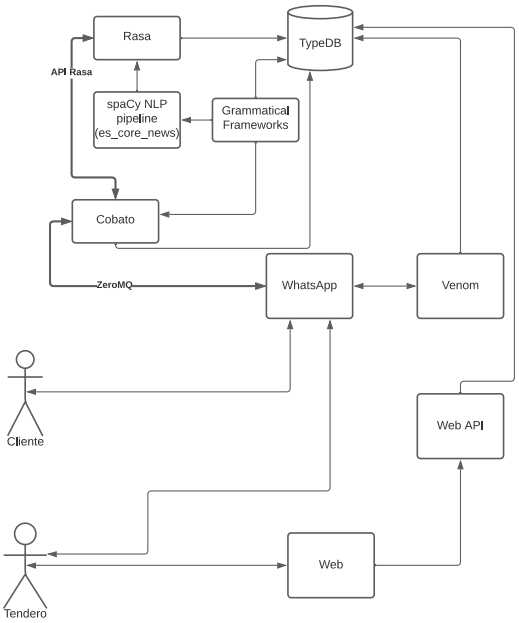

Figura 1: Arquitectura general de COBATO

- Servicios de back-end
  - *Node.js*. Actúa como una capa middleware entre la interfaz web y la base de datos, un grafo de conocimiento implementado en TypeDB.
  - *Python*. Es el lenguaje de programación genérico usado para implementar diversas librerías para comunicar los servicios de PLN, front-end y back-end.
  - *Rasa*. *Framework* para la implementación de asistentes conversacionales basados en texto.
  - *TypeDB*. Gestor de Base de Datos que toma como modelo conceptual de datos el modelo Entidad-Relación, y que se implementa mediante un modelo lógico de datos basado en hipergrafos. A diferencia de otros DBMS basados en hipergrafos, TypeDB requiere de un esquema de datos definido.
- Recursos externos.
  - *Grammatical Frameworks*. Lenguaje de propósito especial que se compila en última instancia en una gramática multilingüe, que consta de una sintaxis abstracta y un conjunto de sintaxis concretas. La sintaxis abstracta define un sistema de árboles sintácticos, y

las sintáxis concretas completan la gramática codificando la correspondiente información morfo-sintáctica, morfológica y léxica, particular de cada lenguaje.

– *Spacy NLP es_core_news*. Modelo de lenguaje usado para el flujo de PLN para procesar la entrada del usuario: extracción de entidades, clasificación de la acción (intención) de usuario, tokenización, similitud semántica de términos, etc.

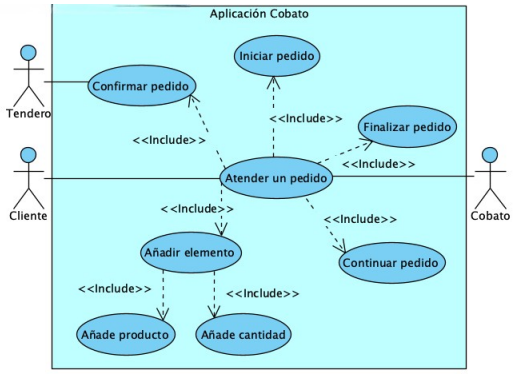

Figura 2: Caso de uso: atender un pedido

3. Un caso de uso: las fruterías

En esta primera versión COBATO se ha adaptado al caso concreto de las fruterías, identificándose diez casos de uso principales, relativo a la elaboración de un pedido. Otros casos contemplados refieren el registro y gestión de contenido de comercio a través de la aplicación web, o gestionar ofertas a través de un canal privado entre el comercio y el chatbot. Tomando al cliente como actor principal destaca el registro del canal de WhatsApp y la gestión de pedidos. En relación al canal de WhatsApp para cada cliente, durante el proceso de registro el comercio obtiene un enlace junto con el código QR equivalente. Cuando un cliente desea interactuar con el chatbot solo necesita escanear con su móvil tal código QR. Automáticamente se crea un canal en el cual son miembros el mismo cliente, el comercio y COBATO. Nótese que, en consecuencia, las interacciones cliente-COBATO son igualmente accesibles por el comercio, que puede intervenir en cualquier momento. Un segundo ejemplo destacado desde la perspectiva del cliente es la atención a un pedido (Figura 2). Se corresponde con la transacción a lo largo de la cual el chatbot solicita al cliente qué productos necesita, y en qué cantidad. Una vez el cliente desee finalizar, la transacción queda en estado "pendiente", hasta que, eventualmente, el comercio confirme el pedido en el mismo canal WhatsApp que comparte con el cliente y COBATO, pasando entonces el pedido a "atendido".

Previo al uso de COBATO por parte del cliente, es necesaria su instanciación y puesta en marcha: la BBDD implementada en TypeDB ha sido instanciada con los productos disponibles (frutas y verduras en este caso). También se añade información léxica, sintáctica y semántica del lenguaje entendido por COBATO, a modo de lenguaje controlado en el ámbito de la aplicación, y que representa las posibles interacciones cliente-chatbot expresadas en un lenguaje natural controlado conforme los casos de uso identificados. Ejemplos de expresiones legales en este lenguaje controlado son "cuál es el horario en fin de semana", "¿cuál es la dirección del comercio?", "añade un kilo de producto " o "¿qué precio tiene producto ", donde "producto " es cualquier de los productos dados de alta en la BBDD, frutas y verduras en este caso. Posteriormente, este lenguaje controlado es traducido a un programa GF, una gramática, mediante un script escrito en Python. Como se indicó en la sección 2, los GF son muy precisos para el análisis semántico conforme la gramática especificada, pero a su vez limita el lenguaje que es posible analizar. Es por ello que usualmente los *frameworks* tales como Rasa utilizan modelos de lenguaje, de corte probabilístico, en aras de alcanzar una mayor cobertura aun a costa de pérdida de precisión. Aún en este escenario los GF son de gran utilidad: mediante un proceso denominado linearización, GF genera todos los posibles árboles gramaticales plausibles conforme el programa GF escrito. De este modo obtenemos un conjunto de ejemplos con el cual ajustar al ámbito concreto de esta aplicación el modelo de lenguaje que usa Rasa (*es_core_news* en este caso), como parte de su flujo de PLN. Particularmente, estos ejemplos están etiquetados con cada una de las acciones de usuario conforme los casos de uso identificados, así como con las entidades relevantes que allí se encuentran, principalmente frutas y verduras. Finalmente, como parte de esta puesta en marcha de COBATO se han escrito diversos "scripts" para Rasa, de modo que este pueda gestionar la lógica de diálogo y generar las respuestas de usuario necesarias. Una vez COBATO está en funcionamiento, el comercio se ha registrado en el sistema, y el cliente ha usado el código QR correspondiente, este último puede empezar a interactuar con COBATO, y con el dependiente, a través de un canal WhatsApp.

Las solicitudes de cliente son, en primera instancia, atendidas por el front-end Venom, el cuál actúa como pasarela entre WhatsApp y Rasa, el framework que finalmente atenderá las solicitudes de usuario. La respuesta que Rasa proporciona al usuario se confecciona conforme la intención de este y el KG almacenado en TypeDB. Para poder identificar la intención de usuario, previamente Rasa es entrenado acorde los guiones escritos a tal efecto y el modelo de lenguaje previamente ajustado con los ejemplos provistos por GF.

4. Conclusiones y trabajo futuro

Se propone el desarrollo de un chatbot cuyo conocimiento se codifica en KG y GF, de modo que toda la información queda en un único repositorio, con las ventajas que ello conlleva desde el punto de vista de la integridad y consistencia de los datos, además de facilitar el mantenimiento, escalado y migración del sistema. El proyecto está en fase de prueba, con lo que el siguiente paso será probarlo en entornos reales. A medio plazo, se pretende avanzar en el uso de los KG. Concretamente, se debe incluir en este conocimiento para la gestión del flujo de diálogo y la generación de las respuestas automáticas. Ambos aspectos actualmente son implementados en Rasa. Separar completamente el *framework* del modelo de conocimiento permitirá el desarrollo de un gestor de diálogo basado íntegramente en grafos.

Agradecimientos

Este proyecto es parcialmente financiado con fondos de la Oficina de Transferencia de Resultados de la Investigación de la Universidad de Jaén y del Instituto de Estudios Giennenses, área de conocimiento de Ciencias Naturales y Tecnología, así como por el Gobierto español a través del proyecto RTI2018-094653-B-C21, LIVING-LANG.

Bibliografía

A. Ranta, Grammatical framework, Journal of Functional Programming 14 (2004) 145–189.

A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, et al., Knowledge graphs, ACM Computing Surveys (CSUR) 54 (2021) 1–37.

A. Fensel, Z. Akbar, E. Kärle, C. Blank, P. Pixner, A. Gruber, Knowledge graphs for online marketing and sales of touristic services, Information 11 (2020) 253.

D. Fensel, U. Şimşek, K. Angele, E. Huaman, E. Kärle, O. Panasiuk, I. Toma, J. Umbrich, A. Wahler, Why we need kg: Applications, in: Knowledge Graphs, Springer, 2020, pp. 94–123.

P. Christmann, R. Saha Roy, A. Abujabal, J. Singh, G. Weikum, Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion, in: 28 CIKM, 2019, pp. 729–738.