Instituto Tecnológico Autónomo de México

Modelo de Lenguaje Condicional: conversaciones para dar órdenes a un asistente virtual

José Jaime San Juan Castellanos/ 183344

Sonia López Rito/ 183349

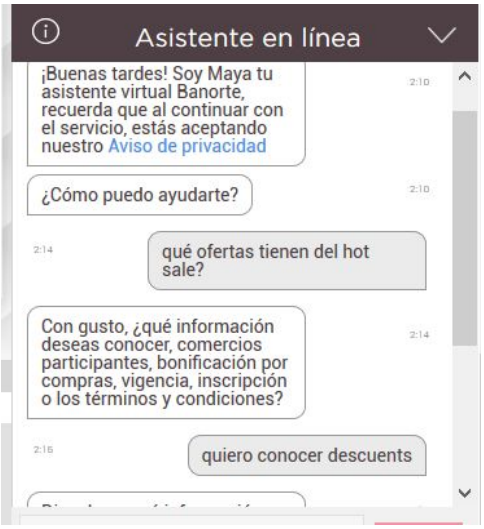
Índice

Modelo de Lenguaje Condicional: conversaciones para dar órdenes a un asistente virtual

Introducción

El procesamiento de lenguaje natural es el área de la inteligencia artificial en la que se analizan y crean modelos para entender el lenguaje del ser humano, y se impulsa el desarrollo de algoritmos y técnicas para que las máquinas puedan reproducirlo de manera exitosa. La tecnología actual nos permite tener reconocimiento óptico de caracteres, reconocimiento automático de voz y traductores de lenguaje con resultados aceptables, así como demás herramientas que, en menor o mayor medida, cuentan con un componente de procesamiento de lenguaje detrás. Una aplicación interesante del procesamiento de lenguaje natural son los asistentes virtuales. Seguramente hemos escuchado hablar de Siri, Alexa, Cortana y otros asistentes que ayudan a las personas a resolver diversas tareas, como poner música, buscar contenido en internet, manejar la agenda, marcar por teléfono a un amigo y otras tareas cotidianas en las que han probado tener éxito y aprobación. Un aspecto vital de los asistentes virtuales consiste en la interacción con el usuario, ya que para que un asistente pueda realizar una tarea, primero necesita comprenderla, es decir, "entender" lo que el usuario está solicitando. Esto puede darse de diversas maneras, por ejemplo, en forma de texto como los asistentes en línea de diversos sitios web, o mediante audio, como algunas tareas que se realizan a través de los teléfonos celulares. En la Figura 1 podemos ver un ejemplo de un asistente virtual de un banco.

Figura 1: Asistente virtual del banco Banorte



Las tareas de procesamiento natural requieren de un modelo de lenguaje, que permita calcular probabilidades y construir predicciones. Asimismo, en el caso de un asistente virtual, se cuenta con una componente adicional, la interacción con el usuario, que genera que el modelo de lenguaje tenga que ser condicionado a la información intercambiada por ambas partes resultado de la interlocución. En los casos en que es necesario generar o predecir una expresión a partir de otra expresión (ambas formadas de una o más palabras) hablamos de un modelo de lenguaje condicional. En este proyecto analizamos un problema de lenguaje condicional, en el cual se utiliza la conversación entre un asistente y un usuario para generar la siguiente expresión del diálogo.

Modelo de lenguaje condicional

Un modelo de lenguaje es una distribución de probabilidades asignada a expresiones o secuencia de palabras. Por su parte, un modelo de lenguaje condicional asigna probabilidades a expresiones, condicionadas a otras expresiones. Este tipo de modelos son de especial aplicación para tareas de traducción e interacción, que se ven reflejadas en bots de solución de problemas, resumen de textos, y en el caso que nos compete en este reporte, el problema de los asistentes virtuales.

En general, un modelo de lenguaje condicional se puede representar como sigue:

$$P_\theta(U_t | f(U_{1:t-1}))$$

donde:

- U_t es la expresión del diálogo en el tiempo t. A su vez, U_t representa un conjunto de palabras $w_{t_1} w_{t_2} \dots w_{t_k}$
- f es una función del diálogo. Por ejemplo, si queremos un comportamiento similar al modelo de n-gramas, esta función podría representar que únicamente consideremos las últimas n expresiones.

Modelos de lenguaje condicional con redes neuronales

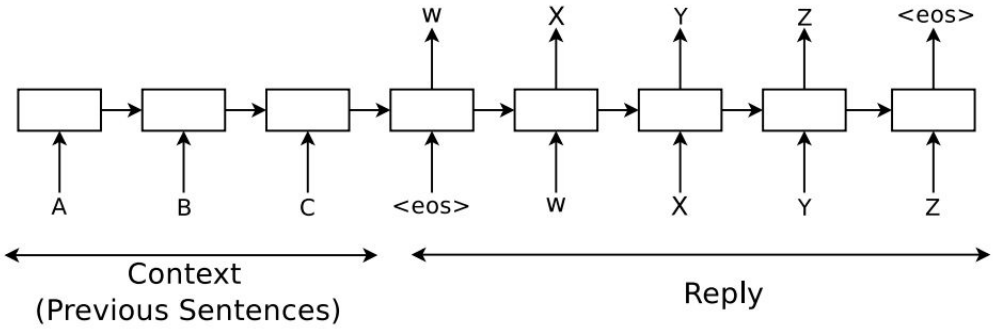
El uso de redes neuronales profundas para el modelos de lenguaje condicional es popular. Por ejemplo, en [2] se describe la manera de entrenar redes neuronales con este fin. La idea principal nació a partir de la popularidad del *framework seq2seq*, una arquitectura marco que permitió modelar problemas relacionados con el mapeo de secuencias a otras secuencias. Las ventajas de este enfoque es que el modelo es autocontenido, en el sentido de que no requiere reglas específicas de lenguaje, sino arquitecturas ya existentes planteadas en el estudio de redes neuronales profundas.

La Figura 2 muestra el siguiente ejemplo: supongamos que tenemos una conversación en la que el usuario menciona la expresión “ABC” y el asistente responde con la oración “WXYZ”. Con esto, la red planteada es de tipo *many-to-many autoencoder*, en la que se utiliza una red neuronal que se entrenará mapeando la expresión “ABC” a la expresión “WXYZ”. En palabras de los autores, utilizar la fortaleza de este modelo recae en su simpleza y generalidad, pues la misma estructura puede ser utilizada para un problema de traducción,

un problema de preguntas y respuestas, así como un modelo de conversaciones sin mayores cambios a la arquitectura. Para un modelo de conversación, la cadena “ABC” sería el contexto y estaría formada por toda la conversación concatenada. La cadena “WXYZ” sería la siguiente expresión de la conversación. Vale la pena señalar que no se ponen restricciones de tamaño sobre las secuencias de entrada o de salida, es decir, son flexibles en cuanto a lo que reciben y a lo que predicen.

También vale la pena señalar los inconvenientes de este enfoque aplicado a la conversación, pues la simplicidad reduce el problema a optimizar una función, mientras que el problema de un lenguaje es que su objetivo es la comunicación, que normalmente es a largo plazo, y se basa en el intercambio de información, más que en la siguiente predicción.

Figura 2: framework seq2seq



Fuente: Tomado de [2]

Descripción de los datos

Los datos utilizados en este proyecto son el conjunto de datos Taskmaster-1, liberado de manera gratuita en 2019 por sus autores (véase [1]). La base de datos está formada por 13,215 diálogos orientados a realizar seis tipos de tareas:

- ➔ Ordenar pizzas.
- ➔ Realizar citas con el mecánico.
- ➔ Solicitar un servicio de transporte (Uber, Lyft).
- ➔ Comprar boletos de cine.
- ➔ Ordenar café.
- ➔ Reservar un restaurante.

El conjunto de datos es artificial, pues fue creado a partir de un experimento controlado con instrucciones generales a fin de lograr el objetivo de contar con conversaciones dirigidas a solicitar/resolver tareas específicas, pero sin restricciones sobre aspectos específicos del lenguaje utilizado, con tal de obtener conversaciones ricas en diversos aspectos de lenguaje, como se espera suceda en la realidad. En palabras de los autores, Taskmaster-1 es un conjunto de datos que proporciona un lenguaje más rico y diverso comparado con los puntos de referencia actuales, ya que se basa en conversaciones sin restricciones y orientadas a tareas que involucran más entidades del mundo real.

Se utilizaron dos procedimientos para crear esta colección de diálogos. El primero involucra un enfoque de "Wizard of Oz" (WOz) hablado por dos personas en el que los trabajadores de crowdsourcing desempeñaron el papel de “usuarios” interactuando con un call center entrenado como “asistentes” (5,507 registros), mientras que el segundo procedimiento fue "self-dialog", en el que los trabajadores de crowdsourcing escriben el diálogo completo ellos mismos desempeñando ambos roles en función de los escenarios descritos para cada tarea (7,708 registros). La primera metodología, no obstante de ser ideal para recrear un escenario real de conversación y así obtener un conjunto de datos robusto, resultó ser compleja, consumió mucho tiempo, además de contar con diversas consideraciones técnicas de implementación y procedimientos administrativos respecto al entrenamiento y control de los trabajadores. Con esto, la decisión de utilizar el segundo procedimiento fue para incrementar la colección y por cuestiones de eficiencia y costo. Un diálogo mediante el segundo método costó en promedio seis veces menos que un diálogo generado mediante el primer procedimiento.

El conjunto de datos contiene observaciones que garantizan la diversidad de los hablantes y la precisión de la conversación debido a que no se restringe a los trabajadores a scripts definidos. Cada conversación tiene la siguiente estructura:

- *conversationId:* identificador único con el prefijo ‘dlg-’.
- *utterances:* serie de enunciados que componen la conversación.
- *instructionId:* referencia a los archivos de instrucciones del usuario (y del agente, en caso de aplicar).

Cada enunciado (*utterances*) contiene los siguientes campos:

- *index:* índice que indica el orden de los enunciados en la conversación.
- *speaker:* indica el rol de quien generó el enunciado, USUARIO o ASISTENTE.
- *text:* texto sin formato del enunciado. En caso de auto diálogo, esto es escrito por el trabajador. En el caso de los diálogos de WOz, se transcriben de las grabaciones habladas de los trabajadores de crowdsourcing.
- *annotations:* matriz de varios segmentos de texto con comentarios semánticos.

Cada comentario (*annotations*) tiene un solo campo *name,* el nombre del comentario.

El conjunto de datos evoca interés en el lenguaje escrito frente al hablado, los patrones de discurso, el manejo de errores y otros fenómenos lingüísticos relacionados con la investigación, el desarrollo y el diseño de sistemas de diálogo.

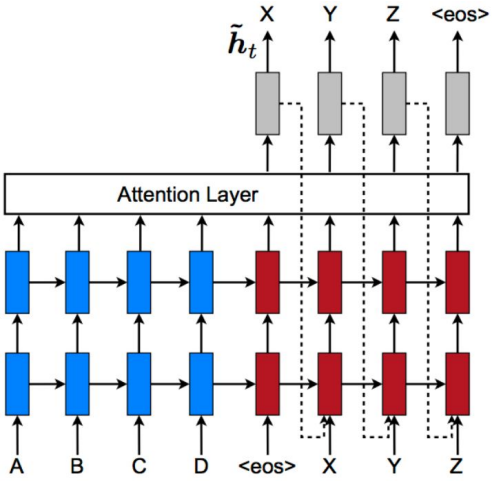
Modelo utilizado

El modelo implementado fue un encoder-decoder con un mecanismo de atención. La parte del encoder es la encargada de abstraer o codificar la entrada en un vector de dimensión fija llamado vector de contexto. El decoder es la capa que se encarga de descifrar el vector de contexto para hacer predicciones precisas. Una parte criticada de los modelos seq2seq es su incapacidad de generar memoria a largo plazo. En algunos casos se ha resuelto esto mediante una arquitectura de red neuronal LSTM a efecto de que la red genere memoria a través del entrenamiento.

El mecanismo de atención fue propuesto por Dzmitry Bahdanau, et al. con la finalidad de resolver la problemática que pudieran causar secuencias muy grandes.

La Figura 3 muestra una representación gráfica de la red utilizada para realizar experimentos.

Figura 3: Encoder-decoder con atención



Observaciones

1.- La limitante de recursos nos obligó a escoger únicamente un tipo de orden (ordenar una pizza). Esto generó un conjunto de entrenamiento de 26,845 pares de diálogos.

2.- La base de datos contiene problemas que afectan profundamente la calidad del experimento. A pesar de que el experimento fue controlado y consiste en analizar el lenguaje de dar órdenes con fines específicos a un asistente virtual, la transcripción generada de los audios a los textos generó diversos problemas inesperados en los datos. a Continuación en la Tabla 1 se muestran algunos ejemplos que presentaron un reto para su limpieza. De igual manera, los typos generaron un problema, pues el modelo de lenguaje asume que la limpieza de la información es parte de otro problema.

Tabla 1: Problemas de la base de datos

<table>
<tr>
<td>

```
{
  "index": 27,
  "speaker": "ASSISTANT",
  "text": "OK. Peace out, JP (*'cause
  we're going to have colloquial
  speech in our AI soon, right?)"
}
```

</td>
<td rowspan="2">

```
{
  "index": 20,
  "speaker": "USER",
  "text": "n/a"
},      {
  "index": 21,
  "speaker": "ASSISTANT",
  "text": "n/a"
},      {
  "index": 22,
  "speaker": "USER",
  "text": "n/a"
},      {
  "index": 23,
  "speaker": "ASSISTANT",
  "text": "n/a"
```

</td>
</tr>
<tr>
<td>

```
{
  "index": 18,
  "speaker": "USER",
  "text": "(25 minutes later) Is someone here?"
},
{
  "index": 19,
  "speaker": "ASSISTANT",
  "text": "Hey here is your pizza."
},
```

</td>
</tr>
</table>

3.- Con tal de generar experimentos ligeros con los recursos a nuestro alcance, en lugar de utilizar un modelo de lenguaje condicional a toda la historia del diálogo, tomamos un enfoque similar a los n-gramas, en el cual mantuvimos en el diálogo únicamente las últimas n expresiones previas ($n = 3,\ 4,\ 5$).

El conjunto de entrenamiento utilizado quedó de la siguiente manera:

Tamaño del historial del diálogo (expresiones)	Longitud diálogo, Longitud respuesta (palabras promedio)
$n = 3$	$22,\ 10$
$n = 4$	$30,\ 10$
$n = 5$	$35,\ 10$

Resultados obtenidos

Asistente y usuario

Entrenando la red considerando respuestas del asistente y del usuario, con n= 3 [Experimento1.ipynb]

Entrada	Predicción
i would like to order a pizza from papa johns	hi , i would like to order a pizza from pizza hut
hi i want to eat some pizza	i need to order a pizza from pizza hut
do you have any promotion	i would like to order a pizza from pizza hut
what is the cost of the extra ingredient	ok , i will order the pizza hut , is that correct \?
what is the delivery time	ok , i will order the pizza for me please

```
Epoch 15 Batch 0 Loss 0.2384
Epoch 15 Batch 100 Loss 0.2377
Epoch 15 Batch 200 Loss 0.2708
Epoch 15 Batch 300 Loss 0.2640
Epoch 15 Batch 400 Loss 0.2848
Epoch 15 Batch 500 Loss 0.3148
Epoch 15 Batch 600 Loss 0.3322
Epoch 15 Loss 0.2711
Time taken for 1 epoch 1519.9763906002045 sec
```

Entrenando la red considerando respuestas del asistente y del usuario, con n= 5 [Experimento2.ipynb]

Entrada	Predicción
i would like to order a pizza from papa johns	i would like to order a pizza from pizza hut \. <end>
hi i want to eat some pizza	i want to order a pizza from pizza hut <end>
do you have any promotion	i would like to order a pizza from pizza hut <end>
what is the cost of the extra ingredient	**i want a large pepperoni and sausage pizza with pepperoni and sausage \? yes , i would like a large pepperoni and sausage pizza with pe [se repite la oración muchas veces]**
what is the delivery time	i would like a large pepperoni and sausage pizza with pepperoni and sausage

```
Epoch 15 Batch 0 Loss 0.3643
Epoch 15 Batch 100 Loss 0.3281
Epoch 15 Batch 200 Loss 0.3573
Epoch 15 Batch 300 Loss 0.3409
Epoch 15 Batch 400 Loss 0.3968
Epoch 15 Batch 500 Loss 0.3941
Epoch 15 Batch 600 Loss 0.3359
Epoch 15 Loss 0.3705
Time taken for 1 epoch 663.51744556427 sec
```

* Esta oración tronó el modelo, la oración se repite un gran número de veces

Solo asistente

Entrenando la red con solo los diálogos del asistente, con n= 3 [Experimento3.ipynb]

Entrada	Predicción
i would like to order a pizza from papa johns	hi , i would like to order a pizza from pizza hut
hi i want to eat some pizza	i will pick it up ?
do you have any promotion	hi , i would like to order a pizza from pizza hut
what is the cost of the extra ingredient	yes , that is correct \.
what is the delivery time	i want to order a pizza from papa johns

```
Epoch 15 Batch 0 Loss 0.3137
Epoch 15 Batch 100 Loss 0.2875
Epoch 15 Batch 200 Loss 0.3279
Epoch 15 Batch 300 Loss 0.3492
Epoch 15 Loss 0.3084
Time taken for 1 epoch 226.01698684692383 sec
```

Entrenando la red con solo los diálogos del asistente, con n= 5 [Experimento4.ipynb]

Entrada	Predicción
i would like to order a pizza from papa johns	i would like to order a pizza from pizza hut \.
hi i want to eat some pizza	i want to order a pizza from pizza hut
do you have any promotion	i would like to order a pizza from pizza hut \.
what is the cost of the extra ingredient	i would like to order a pizza from pizza hut \.
what is the delivery time	i would like to order a pizza from pizza hut \.

```
Epoch 15 Batch 0 Loss 0.4576
Epoch 15 Batch 100 Loss 0.4307
Epoch 15 Batch 200 Loss 0.4688
Epoch 15 Batch 300 Loss 0.3720
Epoch 15 Loss 0.4135
Time taken for 1 epoch 498.0246253013611 sec
```

Conclusiones

Las conclusiones obtenidas con este trabajo fueron las siguientes:

- El procesamiento de lenguaje natural es costoso. En este caso, fue necesario reducir nuestro corpus a únicamente un tipo de orden (pedir pizza). De este modo, junto con la arquitectura de red seleccionada logramos correr los experimentos en la nube (google colab versión gratuita) de manera aceptable. Cada experimento tomaba de 2 a 4 horas en ejecutarse, utilizando un entorno con GPUs y con 12 gigas de memoria Ram. Intentamos correr estos experimentos en computadoras personales, sin embargo en ocasiones con algunos hiperparámetros los recursos no eran suficientes incluso ni para levantar la red.

- Los modelos simples pueden dar resultados aceptables. De los cuatro experimentos mostrados, podemos observar que el caso en que únicamente tomamos tres oraciones de diálogo como historia y que entrenamos solo las respuestas del asistente es el que muestra resultados más prometedores. Sin embargo, incluso dichos resultados son mejorables, razón por la cual quizá sea necesario tunear los hiperparámetros de la red, así como agregar más épocas de entrenamiento y más

poder de procesamiento. Lo anterior redunda en que es necesario tener más recursos para alcanzar resultados aceptables en este tipo de experimentos.

- Los aspectos del lenguaje deben ser analizados cuidadosamente. La parte de la limpieza tomó mucho tiempo y requirió de una revisión casi quirúrgica de algunos registros en particular. Dicha limpieza nos llevó a diversas conclusiones: por una parte, en este caso consideramos que era necesario tomar en cuenta los símbolos de puntuación, pues las preguntas y las pausas tenían un sentido gramatical que intuitivamente favorecen el intercambio de información entre el usuario y el asistente. De igual manera, no consideramos la tentación de incluir todos los errores como los dedazos, pues a fin de cuenta lo que se intenta modelar es un lenguaje “puro”, considerando la parte de la limpieza o la imprecisión del mensaje como en otro tipo de problema.

Referencias

[1] Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Andy Cedilnik, Kyu-Young Kim. “Taskmaster-1: Toward a Realistic and Diverse Dialog Dataset”, International Journal of Digital Content Technology and its Applications. Volume 5, Number 3, pp.126-135, March 2011.

[2] Oriol Vinyals, Quoc V. Le. “A Neural Conversational Model”. Arxiv, 2015.

[3] How Does Attention Work in Encoder-Decoder Recurrent Neural Networks. Sitio web: https://machinelearningmastery.com/how-does-attention-work-in-encoder-decoder-recurrent-neural-networks/ Consultado el 25 de mayo de 2020.

[4] Attention and Memory in Deep Learning and NLP. Sitio web: http://www.wildml.com/2016/01/attention-and-memory-in-deep-learning-and-nlp/ Consultado el 25 de mayo de 2020.