Virtual Real Estate Agent: a system for showcasing houses

CSCI544 Final Paper Assignment

Wenhuang Huang

November 24, 2020

1 A better showcasing experience for house buyers

There are many popular housing websites such as Zillow and Trulia to help people find a house. Those websites provide a list of houses currently on sale with well-retouched pictures and tabular data about their locations, number of rooms, price, and more. However, users might easily get lost and feel overwhelmed, because there is too much information on one single page, making the content difficult to read. The order of house features in figure 1 is so messy that users will have a hard time understanding. As virtual reality and augmented reality technology arises in the last few years, the real estate industry has introduced “virtual tours”, shown in figure 2, where house buyers are able to see houses as if they are there. To further improve the experience, I propose a “virtual real estate agent”, a simplified version shown in figure 3, who can automatically introduce the subjects of houses as users move into different places in real-time. Imagine that you put on VR goggles and immerse yourself in the virtual reality of a house you are interested in, you are free to walk around the house while the virtual agent introduces details of the house to you. For example, as you select a house in San Francisco, a virtual agent will introduce an overview of the house by saying “This single-family home is located at 1286 42nd Ave, San Francisco, CA. It is currently for sale and is listed for $1,050,000. This house has 2 bedrooms, 1 bathroom.” As you move upstairs, the agent will respond, “The upper level offers a large master suite with custom drapes, a walk-in closet, and a closet organizer.” With the help of a virtual real estate agent, buyers can gain a more immersive experience and more directly and quickly grasp the main features of a house without reading a lot of information about it.

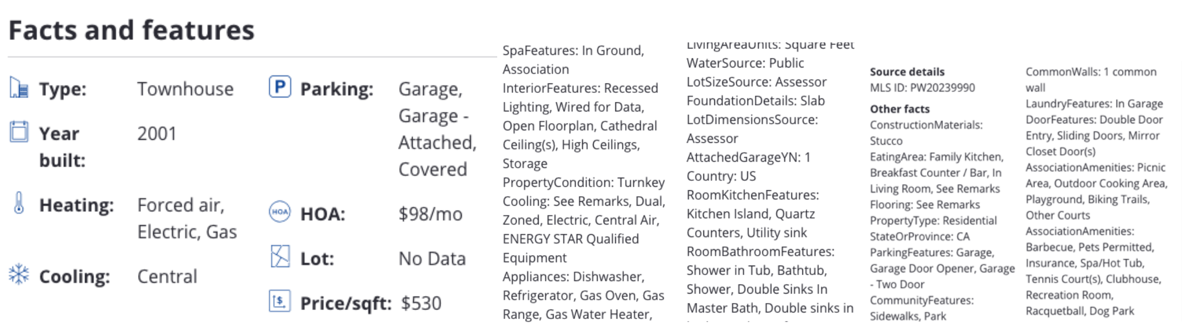

Figure 1. A screenshot on a table of house features from Zillow. The features are poorly written and not well-organized, making the reading difficult.

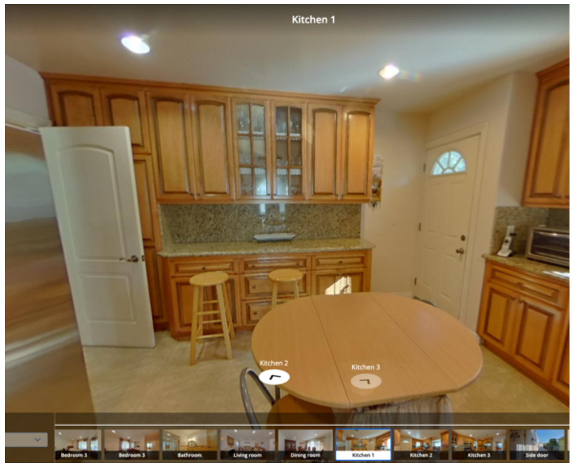

Figure 2. A “virtual tour” from Zillow. Users can click on the living room or the kitchen in order to see different sections of the house.

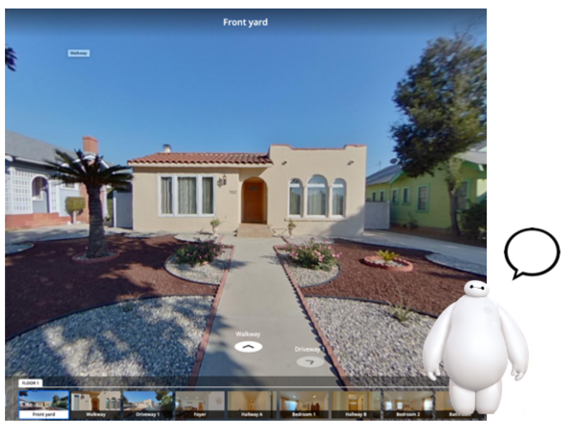

Figure 3. A simplified version of the virtual real estate agent. A virtual character on the right explains details of the house to users.

2 Application as a data-to-text problem

The problem is how to teach our virtual agent to talk, since introducing houses is the major function of the agent. Conversation scripts can be prepared beforehand, but it is not feasible to hire human writers to write all scripts for millions of houses listed on housing websites. I suggest that conversation scripts can be derived from the house data on these websites. For example, when users choose a house, the virtual agent knows that “Eastbluff[neighborhood], 914 Chestnut Place[location]”, and it will respond “This house is situated at 914 Chestnut Place in a highly desirable neighborhood of Eastbluff” based on the data. Hence, the problem becomes how to generate useful text from structured data.

3 Tackle the problem with two different approaches

I propose two approaches to tackle this problem. The first one is a data-driven method that requires human written scripts as the training set. (Wiseman et at., 2017) points out that neural systems for language generation are good at producing fluent outputs but perform poorly at capturing long-term structure. Namely, a neural model has a chance to lose some important information in its final outputs. Therefore, a non-data driven approach is introduced to serve as a baseline for the task.

3.1 Data-driven approach

The model I use is TGen(Dusek and Jurcicek, 2016a), which is based on sequence-to-sequence modeling with attention shown in figure 4. The model consists of an encoder stage which encodes an input sequence, hidden states which are presented by LSTM, and a decoder state which emits the probability over all possible tokens. To make the input as sequences, we concatenate the tabular data into a flat string, tokenize the string, and convert them to their embeddings. On top of this standard seq-to-seq structure, the model implements a beam search for decoding, which keeps the log probability of n possible output sequences. In addition, a classifier to rerank the n best output sequences is implemented, which penalizes those missing required information and/or adding irrelevant ones. Since TGen is open sourced in GitHub, setting up the model is easy. However, we do not have proper datasets to train this seq-to-seq model. Section 4 describes a method to create a dataset. The model will be trained with human written scripts collected from crowdsourcing websites.

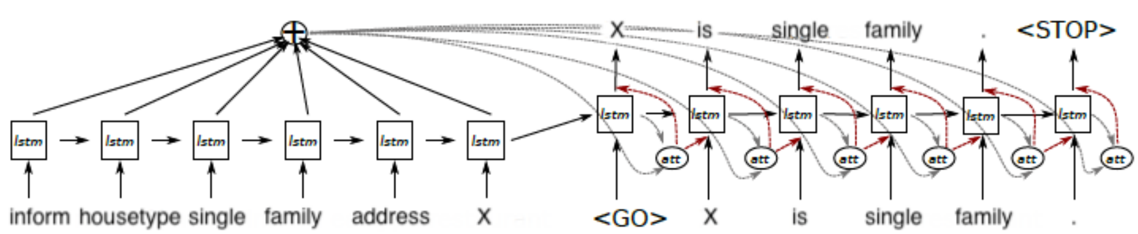

Figure 4. TGen model follows a standard encoder-decoder framework.

3.2 Template-based generator

A template-based generator is intended to solve the problem by filling data in premade templates. The generator takes tabular data as inputs and creates scripts using templatized sentences taken from the training set. When users click on a house, the generator first emits an overview of the house, using templates under the overview category:

> This <houseType> home is located at <address>, built in <yearBuilt>. It is currently for sale and is listed for <price>. <numBedrooms> bedrooms and <numBathrooms> bathrooms and approximately <floorArea> of floor space.

If users click on the living room, the generator fills the slots in templates from the living room category:

> The living room offers <livingRoomFeature>. <lightingFeature> provide a lot of natural lighting.

Each category can have several templates, so the agent emits a slightly different output every time users click.

House Overview		Reference
Attributes	**Example Value**	This single-family home is located at 1286 42nd Ave, San Francisco, CA, built in 1949. It is currently for sale and is listed for $1,050,000. 2 bedrooms, 1 bathroom and approximately 1,097 sqft of floor space.
address	1286 42nd Ave, San Francisco, CA	
houseType	single-family	
floorArea	1,097 sqft	
numBathrooms	2	
numBedrooms	3	
price	$1,050,000	
yearBuilt	1949	
Upper Level		**Reference**
Attributes	**Example Value**	The upper level offers a large master suite with custome drapes, a walk-in closet, and closet organizer. The master bathroom features granite counters, soaking tub, and walk-in tile shower.
masterRoomFeature	walk-in closet, custom drapes, closet organizer	
masterBathroomFeature	granite counters, soaking tub, walk-in tile shower	

Figure 5. Example of the collected dataset. Left: tabular data from housing websites. Right: human written scripts.

4 Method to collect data

The first step is to acquire house features over 50k houses from housing websites like Zillow and Trulia using a web crawler. Data from these websites are not organized in favor of generating text. So I suggest organizing data into different categories such as overview, main level, upper level, and backyard, shown in figure 5. Think of the categories as parts of a house. It is necessary to survey real estate agents to get rid of less important features in a category. The agent will only introduce core attributes in each category. Filtering out houses that have too much missing information to ensure the quality of the dataset.

The second step is to crowdsource the writing for conversation scripts through platforms like Amazon Mechanical Turk or CrowdFlower. Writers will be asked to write colloquial sentences for each category of a house, given the organized tables and a website link to the property. Crowd workers are free to choose the combinations of attributes from the table to create sentences. Link to the property is for assisting their understanding of the data to write better sentences. A descriptive and detailed guideline for the task will be provided to crowd workers. The instruction will include an example of final outputs, a more detailed version of figure 5. Behaviors like copying the sentences and only modifying certain slots associated with tabular data will be penalized. The goal is to gather conversation scripts for 10k houses assuming the payment for the scripts of one house is roughly $1.

5 Evaluation metrics

Automatic metric BLEU is used to evaluate the performances of both approaches. BLEU can give us a sense of how close the outputs to the human written scripts. Additional human evaluation can help to compare the performance of these two approaches. We analyze the readability of these two approaches by asking “does the sentences make any sense”, “how fluent they are on a scale of 1 to 5”, “how many grammar errors in the sentence”, “does the structure of the sentences look weird”. Also, we calculate the percentage of house attributes used to generate texts on both approaches. The higher the rate, the better the approach utilizes the tabular data.

6 Thoughts on modeling

Both data-driven and template-based approach have their pros and cons, and some techniques can be applied in the future to improve them. With a template-based generator, the language is likely to be less fluent and natural, meaning users can easily notice the virtual agent is responding with some premade templates and the language sounds funny when the tabular data do not fit into the templates. What’s more, the generator will have trouble scaling. When new types of data come, more manual work will be involved to create new templates to adapt to the data. The seq-to-seq model has the chance of missing important information in tabular data, as it can decide to retain or ignore previous context. Users might feel confused if the virtual agent does not introduce where the house is and straightly talks about the price in the beginning. As we have more house features, the model can make more mistakes as it “forgets” important features.

Future work can include parsing the input strings to make sure they are in a similar order. This can potentially mitigate the problem of forgetting certain features. Furthermore, the dataset is built with the help of human writers, and it is expensive to acquire text data. As a result, it is difficult to improve the model, simply because of no enough data to train. It will be worth a while to investigate some data augmentation techniques in future works.

7 References

Ondrej Dusek and Filip Jurcıcek. 2016a. Sequence- to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the As- sociation for Computational Linguistics*. Berlin, Germany, pages 45–51. arXiv:1606.05491. http://aclweb.org/anthology/P16-2008.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document gen- eration. *arXiv preprint arXiv:1707.08052*.