Floor Plan Generation and Auto Completion Based on Recurrent Neural Networks

Johannes Bayer
Knowledge Management Dep.
*DFKI**
Kaiserslautern, Germany
johannes.bayer@dfki.de

Syed Saqib Bukhari
Knowledge Management Dep.
*DFKI**
Kaiserslautern, Germany
saqib.bukhari@dfki.de

Andreas Dengel
Knowledge Management Dep.
*DFKI**
Kaiserslautern, Germany
andreas.dengel@dfki.de

***Abstract*—During early design phases, the architect's task is to develop a floor plan layout from a high level description. This process is usually conducted manually nowadays in an iterative manner. In order to assist the architect with repetitive tasks during the individual design steps, we trained a recurrent neural network to mimic the architect's behavior. Our approach is based on sequences that recreate the user's behavior and that we generated from simple floor plans. By utilizing a dedicated inferencing mechanism, we are able to implement the generation of different design steps and tasks using a single LSTM model. We compare two different types of sequencing approaches by calculating their errors on a test set for a selected design step and evaluating the results qualitatively. While the current performance still needs to be improved for productive use, our dedicated inference mechanism shows a functional behavior.**

***Index Terms*—Archistant, Archistant WebUI, LSTM, Early Design Phases, Architectural Support**

I. Introduction

During early design phases, an architect is given a high level description of a building from a costumer (e.g. "Apartment with two sleeping rooms and a 200 square feet living room"). The architect's task is to develop a floor plan layout.

II. Related Work

A. The Long-Short Term Memory

Long-Short Term Memories [1] are a class of recurrent artificial neural networks.

B. Floor Plans as Graphs

Floor plans can be described in a graph-based manner [3]: each room is represented by a node in the graph, each connection between two rooms (wall, door, entrance or passage) is represented as an edge.

III. Proposed Mechanism of Autocompletion of Floor Plans Using LSTM

A floor plan is described by three blocks (each consisting of tags of the same kind): 1. the room function declarations, 2. room connections and 3. the room geometry layouts.

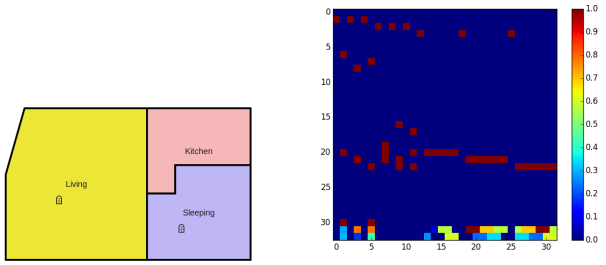

Fig. 1. Left: Rendered Image of a Sample Floor. Window symbols indicate access to natural light. Right: Feature Vector Sequence Encoding of the Same Floor Plan.

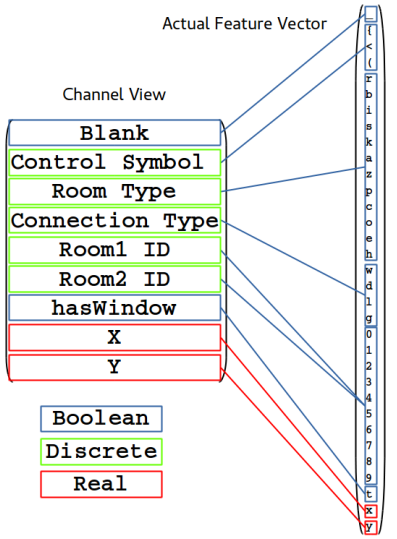

Fig. 2. Structure of the Feature Vector.

A. The Feature Vector

A floor plan has to be described by a sequence of feature vectors for LSTM processing (see fig. 1).

The feature vector (which is used both as the model's input and output) can be considered as structured into several channels (see fig. 2). The components, which encodes a point's

*German Research Center for Artificial Intelligence

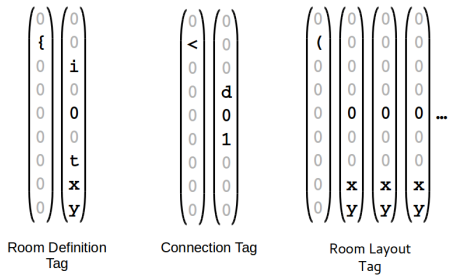

Fig. 3. The Individual Tag Types in Feature Vectors Representation (Channel View). Left: Definition of a living (`i`) room with ID (`1`) with a window (`t`) with a center at position (`x`,`y`). Middle: Definition of a door connection (`d`) between rooms `0` and `1`. Right: Definition of the polygon layout of room `0`.

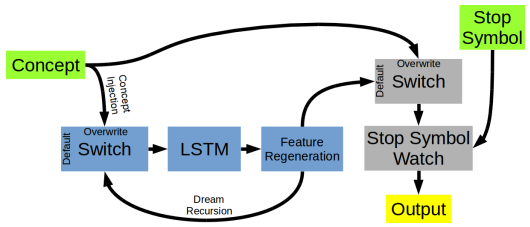

Fig. 4. The ShallowDream structure. The inputs are marked green. Components of the LSTM recursion are marked blue.

position are real values between 0 and 1, all other components are boolean.

B. Tags

A Tag is a series of successive feature vectors representing a design action. A tag starts with a control vector solely indicating the tag's type (see fig. 3). **Room Definition Tags** define the very room by assigning it an ID as well as a room type, a flag indicating whether or not the room has a window, and the position of its center. **Connection Tags** declare the connection between rooms. They consist of the references between the two connection partners and the connection type. **Room Layout Tags** define a polygon surrounding walls around a room.

C. LSTM Input and Output Sequences

In this paper, we examine two different sequencing approaches: In **Block Generation Sequencers**, the first b blocks followed by blank vectors are given to the LSTMs as input. As output, the $b+1$ th block is used surround by blank vectors. To support the entire work flow, multiple models are needed. **Vector Prediction Sequencers** aim to predict a v-th vector of a sequence given a sequence of the first $v-1$ sequence vectors.

D. The shallowDream Structure

While inferencing is simple for block generation sequencers (forward propagation on the concatenation of all input blocks and a sequence of blank vectors), inferencing in vector prediction sequencers is more difficult. Generally, after a the model's output is generated in one time step, it is reinserted as input in the next time step. We extended Alex Graves metaphor of a dreaming person [2] with the one of a person how is combining both dream and information from outside and hence refer to our inferencing structure as *shallowDream* (see fig. 4).

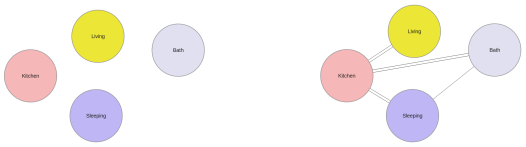

Fig. 5. Left: Input given to the shallowDream structure. Right: Output obtained from the shallowDream structure.

Initially, the existing floor plan (in this context also referred to as *concept*) is fed into the LSTM, simultaneously ignoring the model's output (*concept injection*). After that, the LSTM takes over both the generation of the structures output that also serves as its own input. Using the shallowDream structure, we are able to implement multiple different functions by simply altering the concept and the stop symbol.

E. Performed Functions

Room Connection Generation: Given a set of rooms (center position coordinate and room function), connections are generated between them. **Room Layout Generation**: Given a room Graph, layouts for each room a layout (i.e. a polygon describing its surrounding walls) is generated.

IV. Experiment and Results

We trained LSTMs based on our two sequencing approaches. In all cases, we used a training database with 200 entries, a test database of 40 entries, 500 LSTM cells and a learning rate of 0.01.

A. Quantitative Analysis

We calculated the error both approaches made for the connection generation function on the test set. The results are shown in Table IV-A.

Approach	Error
Block Generation Sequencer	65.78%
Vector Prediction Sequencer	66.08%

TABLE I
Performance comparison of the individual approaches for the connection generation task on the test set.

B. Qualitative Analysis

The performance of the shallowDream structure is shown exemplary for the room connection generation (see fig. 5).

V. Conclusion and Future Work

We have shown the general viability of our approach.

References

[1] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
[2] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.
[3] Langenhan, C. (2015). A federated information system for the support of topological bim-based approaches. *Forum Bauinformatik Aachen*.