AUTODESK. UNIVERSITY

SD322076

A Practical Use of Machine Learning in The AEC Industry

John D'Alessandro
KLH Engineers

Doug Johansing
KLH Engineers

Learning Objectives

- Discover how machine learning can be used as a tool in the AEC industry
- Discern what problems can be solved using machine learning
- Learn how to integrate a concrete use of machine learning into a company's workflow

Description

Many industries have embraced machine learning, but its uses are still widely misunderstood. This class will demystify machine learning by providing a case study on how KLH Engineers, PSC (KLH) used Python to create a machine-learning model to bridge the gap between AutoCAD and Revit in the architecture, engineering, and construction (AEC) industry. This study will illustrate which concrete problems machine learning can solve, such as converting thousands of possible AutoCAD layer names into useable Revit elements by utilizing conversions already made by hand. Attendees will discover how to use their own historical data to enhance their company's process workflow.

Speaker(s)

John D'Alessandro is a software engineer at KLH. Since entering the AEC industry, John has developed several innovative tools within and outside of Revit to help engineers spend less time drafting and more time engineering. He has also developed machine learning tools on Azure using .NET and Python. John holds a Bachelor of Science in computer science from the University of Cincinnati.

Doug Johansing, PE, LEED AP BD+C, is a principal of KLH and serves as a programmer and senior electrical engineer. He has over 15 years of experience in the AEC industry and 10 years working with the Revit and AutoCAD APIs. He leads many database driven programming initiatives related to Revit and process innovation. Doug holds a Bachelor of Science in electrical engineering from the University of Cincinnati.

The Layer Name Translator: How We Use Machine Learning

KLH Engineers is an MEP firm with a dedicated software engineering department. It has developed several tools to make MEP engineering processes more efficient and higher quality so that the MEP engineers can spend less time drafting and more time engineering. Through the development of the Layer Name Translator, a tool that converts AutoCAD layer names into useable Revit elements, the department has helped engineers save time and improve quality without changing existing processes.

The Problem

KLH is 100% Revit, meaning that all of its drawings are modeled in Revit. However, this is not the case for the entire industry, and the firm regularly receives AutoCAD files from architects and other clients. KLH has a process for converting the 2D CAD models to 3D Revit models, but it requires standardized layers.

Steps to Convert

After a CAD model is received, it needs to be "cleaned." This is a process of making sure that the model meets internal standards so that the firm's tools can convert the model automatically. One of these standards encompasses the layer names in the model. This is so that we can differentiate between different layers such as walls and furniture and give them the proper properties in Revit.

Steps to Assign Layers

Each layer in the AutoCAD model must be assigned a phase and a category. This was previously done one at a time. The model manager responsible for the conversion would go through each layer, look at it, and determine the phase and category based on how the layer looked. This would require hours of a skilled designer's time devoted to a relatively simple task. Also, they would take shortcuts that could cause issues in conversion later down the line.

A Possible Solution

A number of rules could be set up for translations based on how the AutoCAD layer is named. For example, if a name has the word "DOOR" in it, we can assign it to one of the door categories. An "N" at the beginning means it is in a new construction phase. And so on. This approach would work, but it has a few issues. First off, there are a large number of edge cases to consider. Architects are not using any standard, so the initial list of rules would be gargantuan. In addition, the rules would have to be constantly amended as new patterns are seen. This means that in addition to the huge initial effort to write the algorithm, it would be costly to maintain as time goes on.

Machine Learning

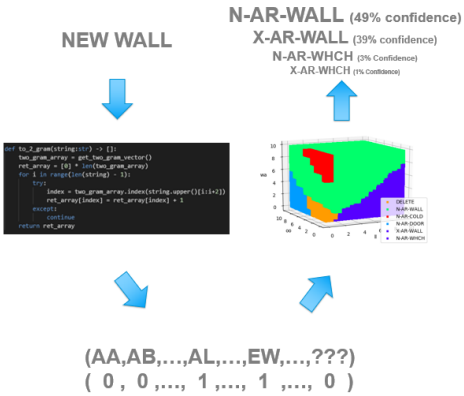
KLH already had 38,000 layer name translations made by its designers over the course of a few years. Instead of writing rules by hand, KLH used this historical data to have a training algorithm write the rules to train a machine learning model.

Technical Effort

The entire new process took a development effort of only 130 lines of Python for the training algorithm and 250 lines of Python to create the server that our AutoCAD routine calls through a REST API. This is because so much of the heavy lifting was done by third party libraries, namely TensorFlow for the training and Flask for the server. Both of these technologies are free to use and the programming effort was minimal. The difficult part of using machine learning was getting the data and meaningfully quantifying it.

Using the Data

A *classification* model was created, meaning its output is one of a number of categories. In this case, the categories were KLH's standard layer names. The training algorithm *softmax* was used because it provides a confidence in each possible result instead of one result. For example, the input "NEW WALL" gave a 49% confidence in "N-AR-WALL," a 39% confidence in "X-AR-WALL," a 3% confidence in "N-AR-WHCH," etc. The inputs to the model have to be numbers, so the challenge was to meaningfully quantify a layer name. Instead of using each letter independently, a *bigram* was made out of the input string. The bigram is a count of each possible pair of letters in the string. For example, "NEW WALL" has 1 occurrence of "WA" but 0 of "DO". The algorithim measures 1,369 combinations, meaning any input can be represented as a point in 1,369-dimensional space. The model, then, can be thought of as a division of that space into KLH's categories.



The Machine Learning Pipeline

When using the machine learning solution to this problem, a model can be trained in about ten minutes. The trained model can be used to almost instantly classify any layer name received. The model doesn't take shortcuts either, meaning that it gives higher quality classifications than the designers that taught it. In addition, it gets increasingly accurate as it constantly receives new data from which it can train.

What Kind of Problems Machine Learning Can Solve

Machine learning is one of many tools to solve problems. It is always important to know when you need a hammer and when you need a screwdriver. There are three major features that our problem had that made it a good problem to tackle using machine learning.

Existing Data

KLH's tools had already been tracking the data about decisions that designers were making. The 38,000 rows of data were instrumental in the development of a solution driven by previous decisions. Without this wealth of history, the model would have been inaccurate and useless. Past attempts at KLH to use machine learning in several other situations failed simply because there was not enough data for the training algorithm to make a meaningful model.

Clear Inputs and Outputs

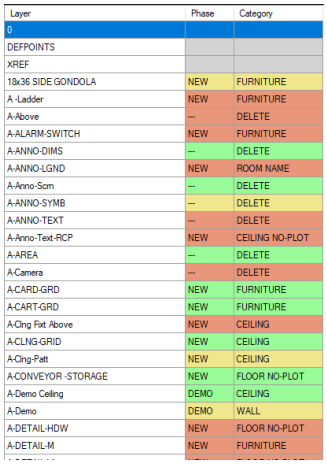
There were clear and precise definitions of what the model would look at and its outputs. Therefore, the data could be easily analyzed and the effectiveness of the model could be measured. Without a defined problem to solve, no data science could give a satisfactory solution.

Patterns Exist Even Though They Aren't Clear

Though there is no explicit standard for how architects name their AutoCAD layers, there are patterns that a model can learn from. If it sees an "LL" in the layer name, it is more confident that that layer represents a wall, because historically architects put "LL"s in the names of their wall layers. Without these patterns, the model would just be guessing. If the patterns were clear, the original approach would have been the more effective solution. This is the most common feature for people to overlook. Either they believe that there are patterns in a chaotic system (such as stock market predictions based off of only stock market data) or the patterns are clear and the machine learning is a waste of resources.

Integrating Machine Learning Into KLH's Process

The layer name translator was a success; however, there were issues along the way due to a lack of professional experience in data analytics. The original plan of having the model inputs be each letter (so "WALL" would look like (87, 65, 76, 76, 0, 0, …)) was flawed. In addition, the users needed to know how confident the model was in its decisions so that they knew which layers to check and correct. Colors were added to represent this confidence (green layers have the highest confidence; red layers have the lowest confidence).



Layer	Phase	Category
0		
DEFPOINTS		
XREF		
18x36 SIDE GONDOLA	NEW	FURNITURE
A -Ladder	NEW	FURNITURE
A-Above	---	DELETE
A-ALARM-SWITCH	NEW	FURNITURE
A-ANNO-DIMS	---	DELETE
A-ANNO-LGND	NEW	ROOM NAME
A-Anno-Scm	---	DELETE
A-ANNO-SYMB	---	DELETE
A-ANNO-TEXT	---	DELETE
A-Anno-Text-RCP	NEW	CEILING NO-PLOT
A-AREA	---	DELETE
A-Camera	---	DELETE
A-CARD-GRD	NEW	FURNITURE
A-CART-GRD	NEW	FURNITURE
A-Clng Fixt Above	NEW	CEILING
A-CLNG-GRID	NEW	CEILING
A-Clng-Patt	NEW	CEILING
A-CONVEYOR -STORAGE	NEW	FLOOR NO-PLOT
A-Demo Ceiling	DEMO	CEILING
A-Demo	DEMO	WALL
A-DETAIL-HDW	NEW	FLOOR NO-PLOT
A-DETAIL-M	NEW	FURNITURE

The Layer Name Translator UI.

Lastly, there were issues with data validation. The training algorithm was reporting that the model was only 80% accurate. However, users said it was almost perfect after they started using it internally. After digging, it was discovered that the model's correct decisions were being scored against incorrect answers. From this experience, here are three tips for anyone wanting to integrate machine learning into their process:

Know What You Want

Don't use a technology for its own sake. Apply tools to the problems they are designed to solve. Machine learning is a tool that is good for a specific set of problems. Keep it in your toolbox but use whatever works best for your desired process. Make sure your problem meets the criteria laid out above before throwing a training algorithm at it.

Keep All Of Your Data

This project would have been impossible if it weren't for the data KLH was already collecting. The data can be massaged and formatted down the line when a use for it comes up, but it will be much easier to store data now than to generate it later for that use.

Use What's Out There

Unless you're an academic, all of the hard parts of machine learning have been done for you. Find tools and information related to whatever problem you're trying to solve. KLH used free technologies and techniques that were found through research to get what was needed. Don't overcomplicate your problem by reinventing the wheel.