ISSN 2320 - 2602
Volume 2, No.10, October 2013
International Journal of Advances in Computer Science and Technology
Available Online at http://warse.org/pdfs/2013/ijacst032102013.pdf

Alternative Design Exploration using K-Nearest Neighbor Technique and Semantic Web Technology in an Energy Simulation Tool

Iman Paryudi[1,2]
[1]Vienna University of Technology, Austria, paryudi@ifs.tuwien.ac.at
[2]Pancasila University, Indonesia, paryudi@gmail.com

ABSTRACT

An energy simulation tool is a tool that is used to calculate energy demand of a building. The existing energy simulation tools carry out alternative design exploration using optimization method. This method works by varying its parameters to obtain better energy performance. The method needs to calculate energy performance every time each parameter is changed. This practice causes the method is slow. Therefore, new techniques to carry out alternative design exploration are used, they are: K-Nearest Neighbor Technique and Semantic Web Technology. The advantage of the above techniques is that they do not need to calculate energy performance for any parameter combination. Instead they will select parameter combinations that will give better design. Experiment shows that, in an alternative design exploration, Semantic Web performs better than KNN in terms of speed. Another advantage of Semantic Web is that there is no need to preprocess data.

Key words : Classification method, K-Nearest Neighbor, Energy Simulation Tool, Semantic Web, Ontology

1. INTRODUCTION

With the issue of EU Directive 2009/91/EC that requires member states to calculate the energy consumption of buildings before the erection of the building [1], all construction projects in all EU countries must be preceded by calculation of energy consumption of the building. Only buildings whose energy performances comply with the regulation are allowed to be built. The calculation is usually carried out by means of energy simulation tools.

Currently, there are a lot of such tools available in the market. In the US, there are more than 389 tools in 2010 [2 in 3]. Meanwhile, there are at least 6 tools in Austria itself.

The existing energy simulation tools carry out alternative design exploration using optimization method. This method works by varying its parameters to obtain better energy performance. The method needs to calculate energy performance every time each parameter is changed. This practice causes the method is slow.

For the reason above, alternative techniques to carry out alternative design exploration are used. I will use K-Nearest Neighbor (KNN) technique and Semantic Web technology. The advantage of the above techniques is that they do not need to calculate energy performance for any parameter combination. Instead they will select parameter combinations that will give better design. By doing this, the number of energy performance calculation is greatly reduced. This will accelerate the alternative design searching process.

This paper will describe the comparison between KNN and Semantic Web in providing alternative design in an energy simulation tool.

The rest of the article will be structured as follows: Section 2 describes the methods used. Section 3 talks about data preparation. Section 4 discusses the result. Section 5 will conclude the paper.

2. THE METHODS

2.1 K-Nearest Neighbor

Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data [4].

The *k*-nearest neighbor algorithm (KNN) is a method for classifying objects based on closest training examples in the feature space. The KNN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification [5]. In this technique, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its *k* nearest neighbors (*k* is a positive integer, typically small).

The algorithm of brute force KNN is as follows:

- Define the number of nearest neighbor k.
- Calculate distance between the query instance and all training samples.
- Sort the training data based on the distance.
- Find the k nearest neighbors.
- Use simple majority of the category of nearest neighbors as the prediction value of the query instance.

Commonly, the Euclidean or the Manhattan distance is used but any other distance can be used instead such as the Chebyshev norm or the Mahalanobis distance [6]. In this experiment, Euclidean distance is used. Suppose the query instance have coordinates (a, b) and the coordinate of training sample is (c, d) then square Euclidean distance is:

$$d^2 = (c - a)^2 + (d - b)^2 \qquad (1)$$

2.2 Semantic Web Technology

Semantic web is a technology that first introduced by Tim Berners-Lee. It consists of a set of technologies, tools, standards which is often represented as layered architecture. Every layer in this architecture can access the functionality of the layers below. The architecture consists of, among others, ontology and querying.

Ontology is a language which can formally describe the meaning of terminology used in Web documents [9]. The current widely used ontology language is Web Ontology Language (OWL). OWL adds capabilities that are not provided by the lower layers. It adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality, equality, characteristics of properties (e.g. symmetry), and enumerated classes [9]. There are three species of OWL: OWL Lite, OWL DL, and OWL Full [10].

In Semantic Web, querying is done by means of SPARQL. It is a query language for RDF. It contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. SPARQL also support aggregation, sub queries, negation, creating values by expressions, extensible value testing, and constraining queries by source RDF graph. The results of SPARQL queries can be result sets or RDF graphs [11].

3. DATA PREPARATION

In classification method, training set is needed to construct a model. This training set contains a set of attributes with one attribute being the attribute of the class. Then the constructed model is used to classify an instance.

For this experiment, there are more than 67 millions of data available. This data comes from combination of 13 parameters with each parameter has 4 possible values (4^{13} data). Since the data is very big, representative training set must be selected. Besides that the training set must be as small as possible. With the above considerations in mind, 5 candidate training sets created. They are with different number of data. The candidate training sets are:

- Training set 1: 1804 data
- Training set 2: 3317 data
- Training set 3: 4382 data
- Training set 4: 5796 data
- Training set 5: 7607 data

To select the best training set, an experiment is carried out. The experiment is done by means of Weka data mining software. It is a software from University of Waikato, New Zealand. Before starting the experiment, the data need to be scaled first. This is to make sure that the data are in the same scale. The scaling is needed because KNN is a classification technique based on distance measure and the distance calculation is influenced by the scale of the data. The scaling process is carried by using min-max method.

For the training set selection, two experiments are carried out. One experiment uses 10-fold cross validation and the other uses 50% training-test sets split. The experiment results are depicted in Figures 1 and 2.

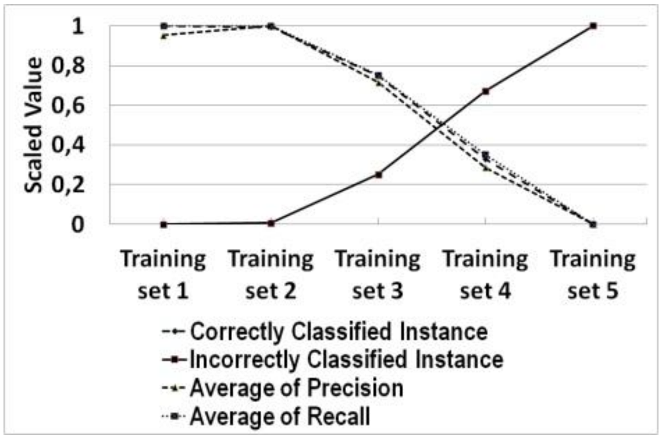


Figure 1. Classifier performance on different training sets using 10-fold cross validation.

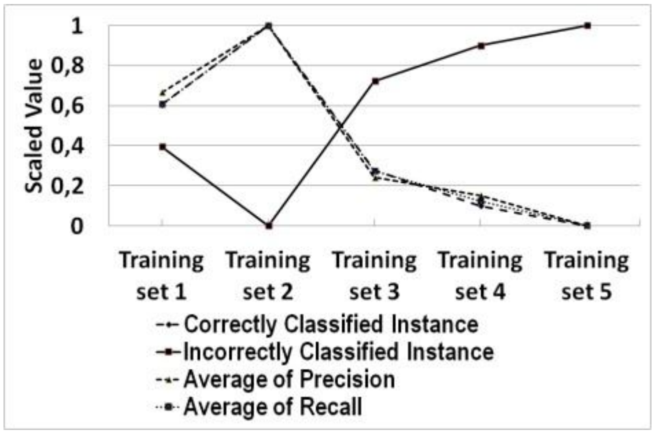


Figure 2. Classifier performance on different training sets using training-test sets split.

Figure 1 shows training set 1, training set 2, training set 3, training set 4, and training set 5 on x axis and scaled value on y axis. It shows that training sets 1 and 2 are two best training sets indicated by high correctly classified instance, average precision, and average recall or low incorrectly classified instance. However, training set 2 is slightly better than training set 1. This is confirmed by Figure 2 showing bigger differences between the two training sets. Based on this result, training set 2 is selected as the working training set.

KNN is a classification technique based on k numbers of neighbors. Hence it is important to select the right k value. Because of that, another experiment is carried out to select the best k value. Since training set 2 is already selected as the working training set, the experiment will use training set 2 with different k values. There are 5 candidate k values: 11, 21, 31, 41, and 51. The experiment is done using 10-fold cross validation and 50% training-test split. The results can be seen in Figures 3 and 4 respectively.

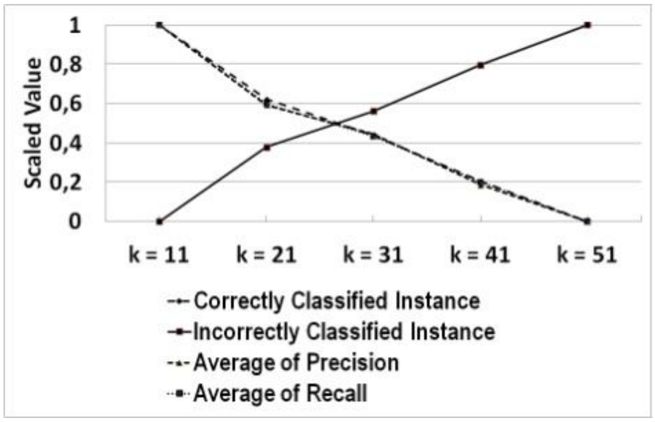


Figure 3. Classifier performance on difference k values using 10-fold cross validation.

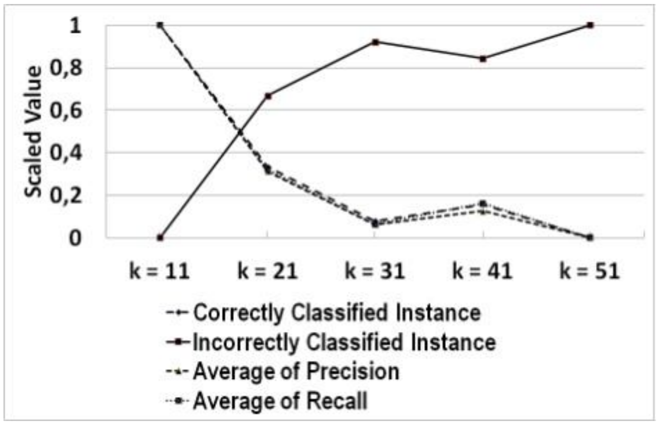
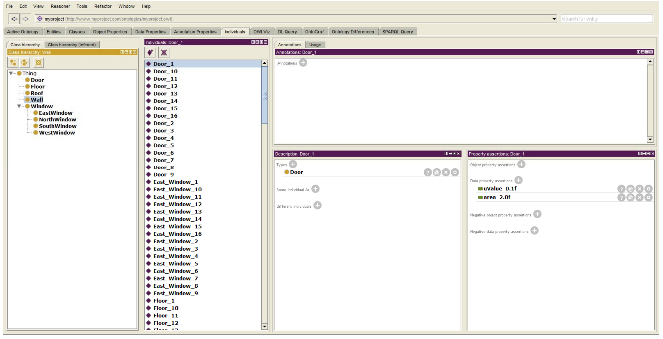


Figure 4. Classifier performance on different k values using training-test sets split.

Both figures show a trend of performance decrease with the k value increase. In other words, the bigger the k value, the worse the performance. The same conclusion was also stated by [7] where they concluded that modifying the number of neighbors did not result in higher accuracy. Moreover Han, Karypin, and Kumar [8] also stated that the bigger the value of k, the worse the accuracy. This result justifies the selection of 11 for the working k value.

Based on the above result, training set 2 and k value of 11 will be used in the KNN classification.

To work with Semantic Web, ontology must be first created. In this case, ontology is created using Protégé (Figure 5). The ontology is then sent to Jena to be processed further. In the processing, a query will be carried out. The query is done by means of Jena ARQ (Figure 6).



Thing
Door
Floor
Roof
Wall
Window
EastWindow
NorthWindow
SouthWindow
WestWindow

Door_1
Door_10
Door_11
Door_12
Door_13
Door_14
Door_15
Door_16
Door_2
Door_3
Door_4
Door_5
Door_6
Door_7
Door_8
Door_9
East_Window_1
East_Window_10
East_Window_11
East_Window_12
East_Window_13
East_Window_14
East_Window_15
East_Window_16
East_Window_2
East_Window_3
East_Window_4
East_Window_5
East_Window_6
East_Window_7
East_Window_8
East_Window_9
Floor_1
Floor_10
Floor_11
Floor_12

Types: Door

uValue 0.1f
area 2.0f

Figure 5. Ontology created using Protégé for the Semantic Web experiment.

```
public void wall(Model model){
        Query query = QueryFactory.make() ;
        query.setQuerySelectType() ;

        ElementGroup elg = new ElementGroup() ;

        String BASE =
"http://www.myproject.com/ontologies/myproject.owl
#";
        Var varIndividu = Var.alloc("individu") ;
        Var varVal = Var.alloc("val") ;
        Var varHeight = Var.alloc("height");

        String className = "Wall";
        float uValue = wallUValue;
        float height = wallHeight;

        Triple t1 = new Triple(varIndividu,
RDF.type.asNode(), Node.createURI(BASE+className))
;
        elg.addTriplePattern(t1) ;

        Triple t2 = new Triple(varIndividu,
Node.createURI(BASE+"uValue"), varVal);
        elg.addTriplePattern(t2);

        Expr expr = new E_LessThanOrEqual(new
ExprVar(varVal), NodeValue.makeNodeFloat(uValue))
;
        ElementFilter filter = new  ElementFilter(expr) ;
        elg.addElementFilter(filter) ;
```

```
      Triple t3 = new Triple(varIndividu,
Node.createURI(BASE+"height"), varHeight);
      elg.addTriplePattern(t3);

      Expr expr2 = new E_Equals(new
ExprVar(varHeight),
NodeValue.makeNodeFloat(height)) ;
      ElementFilter filter2 = new  ElementFilter(expr2) ;
      elg.addElementFilter(filter2) ;

      // Attach the group to query.
      query.setQueryPattern(elg) ;

      // Choose what we want
      query.addResultVar(varIndividu) ;
      query.addResultVar(varVal);
      query.addResultVar(varHeight);

      // Create PREFIX
      query.getPrefixMapping().setNsPrefix("rdf" ,
RDF.getURI()) ;
      query.getPrefixMapping().setNsPrefix("myproject",
BASE) ;

      QueryExecution qexec =
QueryExecutionFactory.create(query, model) ;

      try {
         ResultSet rs = qexec.execSelect() ;

         for ( ; rs.hasNext() ; )
         {
            QuerySolution rb = rs.nextSolution() ;
            float uvalue = rb.getLiteral("val").getFloat();
            wallData.add(uvalue);

            float hei = rb.getLiteral("height").getFloat();
            wallData.add(hei);
         }
      }
      finally
      {
         qexec.close() ;
      }
   }
```

Figure 6. Example query using Jena ARQ.

4. RESULT AND DISCUSSION

Using the data prepared above, experiments using both methods are carried out. Five alternative design searches are done and the process times are recorded. The experiments give the following result:

- Average process time of KNN is 23798 milliseconds
- Average process time of Semantic Web is 705 milliseconds

The result shows that alternative design search using Semantic Web is about 24 times faster than using KNN. This might be caused by the fact that in KNN the data is saved in a database meanwhile the ontology is saved in the form of a file. Accessing database needs more time than accessing a file which is saved in memory. Another reason is that KNN uses classification to search alternative design. Meanwhile Semantic Web uses query. In this case query is much simpler than classification hence faster.

Besides the better performance, there is another advantage of Semantic Web over KNN. In KNN, we need to do preprocessing step before doing the classification. As explain above, we must scale the data, select representative training set, and choose k value. There is no such process in the Semantic Web.

5. CONCLUSION

When applied in alternative design exploration in an energy simulation tool, Semantic Web does the job faster than KNN. Besides that, Semantic Web has additional advantage that is we do not need to scale data, select training data, and choose k value as we must do in KNN.

ACKNOWLEDGEMENT

I would like to thank Directorate General of Higher Education, Ministry of Education and Culture, Republic of Indonesia for the scholarship awarded to me.

REFERENCES

1. S. Dimas. **Commission Directive 2009/91/EC**, Official Journal of the European Union, L 201, 39 – 41, 2009.
2. DOE. **Building Energy Software Tool Directory**, http://apps1.eere.energy.gov/buildings/tools_directory/
3. S. Attia. **State of the Art of Existing Early Design Simulation Tools for Net Zero Energy Buildings: A Comparison of Ten Tools**, http://www-climat.arch.ucl.ac.be/s_attia/attia_nzeb_tools_report.pdf.
4. Oracle. **Classification. Oracle Data Mining Concepts**, 11g Release 1(11.1), Part Number B28129-04 (2005).
5. Wikipedia. ***k*-nearest neighbor algorithm**, http://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm
6. V. Garcia, C. Debreuve. **Fast k Nearest Neighbor Search using GPU**, IEEE, 2008.
7. M. Pazzani, D. Billsus. **Learning and Revising User Profiles: The Identification of Interesting Web Sites**, Machine Learning 27, pp. 313 – 331, 1997.
8. E-H. Han, G. Karypis, and V. Kumar. **Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification**, Springer Verlag, Berlin, 2001.
9. W3C. **OWL Web Ontology Language Overview**, http://www.w3.org/TR/owl-features/
10. G. Antoniou, F. v. Harmelen**. A Semantic Web Primer**, The MIT Press, Cambridge, Massachusets, 2004.
11. W3C. **SPARQL 1.1 Query Language**, http://www.w3.org/TR/sparql11-query/