



POLITECHNIKA WARSZAWSKA

WYDZIAŁ MATEMATYKI
I NAUK INFORMACYJNYCH



PRACA DYPLOMOWA MAGISTERSKA
INFORMATYKA

Retrieval and clustering of real estate advertisements

Wyszukiwanie i grupowanie ogłoszeń
nieruchomości

Deepak Thukral

Promotor: dr. Maciej Grzenda

Warsaw November 2009 MMIX

Abstract

Real estate properties for sale or rent are frequently available through different real estate agencies and published on different web sites. As a consequence, differences in price and description can be also observed. At the same time, it is interesting for a potential buyer or renter to get the entire description of a property of interest based on the data available on different portals and if possible track the changes in price.

One of the problem, faced is data is often incomplete. Hence, standard clustering techniques are not sufficient. In this work, we proposed a novel technique to cluster incomplete data. This method is based on marginalization and weight.

This technique is compared with some of other existing techniques. This work is concluded with possible future work and extensions to the propose solution.

Keywords: *Data Mining, Clustering, Information Retrieval*

Acknowledgement

Foremost, I would like to thank my supervisor dr. Maciej Grzenda for the suggestions and advice he gave me. I further extend my gratitude to his in-depth interest in this work and encouragement of my decisions in this work. Also, I would like to thank everyone in MiNI, who helped me out to focus on the thesis.

I am deeply indebted to my parents, for their love and moral support during my studies. I would also like to thank my friends, for their continuous moral support. I would like to give special thanks to all of them who have been supporting me, without whom it would not have been possible to complete this work.

Contents

List of Figures	viii
List of Tables	x
1 Introduction	1
1.1 Motivation	2
1.1.1 Information Retrieval	2
1.1.2 Data Clustering	3
1.1.3 Components of Clustering Task	6
1.2 Goals	7
1.3 Assumptions	8
1.4 Clusty - An Information Retrieval system	8
1.5 Overview	9
2 Background and Related Work	10
2.1 Web Crawler	10
2.1.1 Crawling Policies	11
2.1.2 Crawler Identification	13
2.2 Latent Semantic Analysis	13
2.2.1 The Classic Vector Space Model	13

2.2.2	Latent Semantics Indexing	14
2.3	Clustering	15
2.3.1	k-means Clustering	15
2.3.2	Quality Threshold Clustering	17
2.3.3	Canopy Clustering	19
2.3.4	Fuzzy C-Means Clustering	21
2.3.5	Locality-Sensitive Hashing based clustering	23
2.4	Real Estate websites	24
2.5	Problems	25
2.6	Related work	26
3	Algorithms	28
3.1	Feature selection	28
3.2	Clustering Algorithms	29
3.2.1	Distance function	30
3.2.2	Weighted Clustering	31
3.2.3	Weighted+LSH Clustering	32
4	Implementation	35
4.1	Clusty - Introduction	35
4.2	General Architecture	37
4.3	Development Environment	40
4.4	System Overview	40
4.4.1	Crawling Agents	41
4.4.2	Text Processing Component	42
4.4.3	Document Repository	45

- 4.4.4 ORM and Query Analyzer 46
- 4.4.5 Clustering Component 49
- 4.4.6 Context Rendering Component 51
- 4.5 Application User Interface 52
 - 4.5.1 Main Screen 52
 - 4.5.2 Result Screen 53
 - 4.5.3 Filter Screen 53
 - 4.5.4 Detail Screen 56
 - 4.5.5 Cluster Screen 58
 - 4.5.6 Trend Screen 59
 - 4.5.7 Admin parameter settings 60
 - 4.5.8 Admin Management Screen 60

5 Evaluation 62

- 5.1 Unit Testing 62
- 5.2 Testing Environment 64
- 5.3 Reference classes 65
- 5.4 Results 65
 - 5.4.1 k-means 65
 - 5.4.2 FCM 68
 - 5.4.3 minHash (LSH) 69
 - 5.4.4 Weighted clustering 70
 - 5.4.5 FCM+LSH 71
 - 5.4.6 Weighted+LSH 72
- 5.5 Comparison Plot 72

<i>CONTENTS</i>	vii
6 Conclusion and Future Work	74
6.1 Conclusion	74
6.2 Future Work	75
References	77

List of Figures

1.1	Data Clustering	4
2.1	K-means illustration	18
2.2	QT illustration	19
2.3	An example of canopy clustering: showing four data cluster canopy covering them. C and D are in same canopy, A, B, E are other three canopies.	20
4.1	Conceptual main units of clusty	36
4.2	Detailed view of clusty units and how they interact	38
4.3	Clusty Database Schema	39
4.4	Detailed view of Crawling agents	41
4.5	Detailed view of Text Processing component	43
4.6	Detailed view of Data repository	45
4.7	Detailed view of ORM and query component	48
4.8	Detailed view of clustering component	50
4.9	Detailed view of context rendering component	52
4.10	Main Screen	53
4.11	Results Screen	54

LIST OF FIGURES

ix

4.12 Filter Screen	55
4.13 Detail Screen	56
4.14 Clustered item on detail screen	57
4.15 Clustered screen	58
4.16 Trend screen	59
4.17 Clustering parameters UI	60
4.18 Admin management screen	61
5.1 Final Results	73

List of Tables

1.1	Difference between Data and Information Retrieval	3
2.1	Comparison of various clustering algorithms	24
4.1	An Example of an inverted index	46
4.2	An Example of how queries are translated	49
5.1	k-mean EMP Contingency table	66
5.2	k-mean NNMP Contingency table	66
5.3	k-mean performance on dataset	67
5.4	FCM EMP Contingency table	68
5.5	FCM NNMP Contingency table	68
5.6	FCM performance on dataset	68
5.7	LSH EMP Contingency table	69
5.8	LSH NNMP Contingency table	69
5.9	LSH performance on dataset	69
5.10	Weighted EMP Contingency table	70
5.11	Wighted NNMP Contingency table	70
5.12	Weighted clustering performance on dataset	70

LIST OF TABLES

xi

5.13 FCM+LSH EMP Contingency table	71
5.14 FCM+LSH NNMP Contingency table	71
5.15 FCM+LSH performance on dataset	71
5.16 Weighted+LSH EMP Contingency table	72
5.17 Weighted+LSH NNMP Contingency table	72
5.18 Weighted+LSH performance on dataset	72

Chapter 1

Introduction

World Wide Web (WWW) is a vital source of information. Mining useful information in WWW has been a strong point of research in last decade. Since expansion of WWW there have been lots of research in information retrieval (IR) and data mining. Data clustering has been strenuously used computer scientists to minimize efforts in processing huge information from the web. Data clustering is useful in many tasks [JMF99].

- Data Mining
- Information Retrieval
- Image Segmentation
- Genes expression patterns [YR01]

According to a major search engine Google around 36% of web pages on Internet are exact or near replica [BP98] mostly due to multiple source of single information on WWW for example, News articles, real-estate listings,

user profiles, etc, Hence there is a need of exact and near replica detection, furthermore in some cases it can be used to track information development and recommendation.

1.1 Motivation

1.1.1 Information Retrieval

The meaning of the term *Information Retrieval* (IR) can be very broad. Just getting a credit card out of your wallet so that you can type in the card number is a form of information retrieval. However, as an academic field of study, information retrieval might be defined thus:

Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers). [MRS08]

Data retrieval, document retrieval, text retrieval and information retrieval are related to each other, but each of them has its own domain, theory, literature and technologies. IR is interdisciplinary, based on mathematics, library science, information science, computer science, physics, statistics, linguistics and cognitive psychology.

Table 1.1 shows some of the distinguishing properties of data and information retrieval, which differentiate between data retrieval (DR) and information retrieval (IR). Information retrieval systems are used to suppress what has been called *information overload*¹. IR systems is being used by universities and public libraries use IR systems to provide access to books,

¹Excess amount of information mostly redundant

	Data Retrieval (DR)	Information Retrieval (IR)
Matching	Exact match	Partial match, best match
Inference	Deduction	Induction
Model	Deterministic	Probabilistic
Classification	Monothetic	Polythetic
Query language	Artificial	Natural
Query specification	Complete	Incomplete

Table 1.1: Difference between Data and Information Retrieval

journals and other documents. Web search engines like Google, Yahoo, Bing are the most visible IR applications. Instead of just text indexing, nowadays IR also includes Information Filtering (IF), Information Classification, Data clustering, etc.

Google - A giant search engine use data clustering and link analysis in IR system to cluster similar web pages [BP98]. Google News is another application where Google uses similar approach to cluster similar news articles. Furthermore, its recommendation system also uses data clustering.

Data clustering techniques can be used to track information. Google news uses clustering techniques to cluster similar news article on data stream from various sources [DDGR07]. This method can be used to generate information timeline from multiple sources clustered together.

1.1.2 Data Clustering

Data clustering or analysis is an unsupervised classification of data into groups so that data within the same cluster share similar properties than data in different clusters.

Clustering (Unsupervised classification) is conceptually different from *dis-*

criminant analysis (supervised classification). In supervised classification *training* is provided and patterns is being used to learn the description of classes which can be used to label new pattern. However, in clustering, the problem is to group a given collection of unlabelled patterns into meaningful cluster, hence its *data driven*; that is, they obtained softly from the data itself [JMF99].

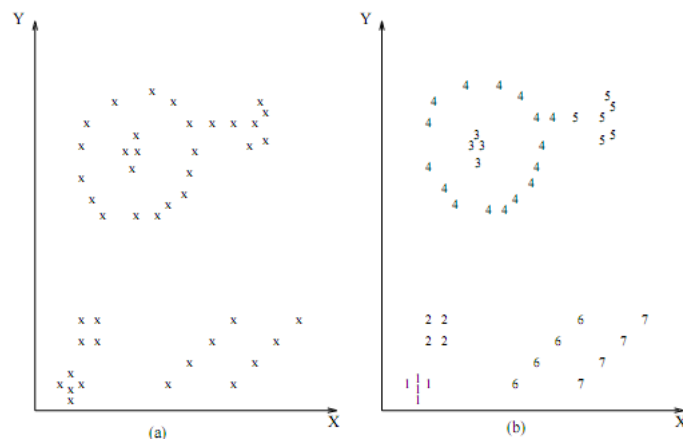


Figure 1.1: Data Clustering

Figure 1.1 depicts a typical example of data clustering of unlabelled data. The input patterns are shown in Figure 1.1(a) and clusters are shown in Figure 1.1(b), in this example points belonging to same clusters are given same label.

An important step in any clustering method is to select *distance function*. A distance function determines the distance between two elements. This drastically changes the shape of cluster. For example, in a 2-D, the distance between the point $(x = 1, y = 0)$ from origin $(x = 0, y = 0)$ is always 1, according to the usual norms, but the distance between the point $(x =$

1, $y = 1$) and the origin can be 2, $\sqrt{2}$ or 1, if you take respectively the 1-norm, 2-norm or infinity-norm distance. Some of the common distance functions are the following:

- The Euclidean distance (also called distance as the crow flies or 2-norm distance).
- The Manhattan distance (aka taxicab norm or 1-norm)
- The maximum norm (aka infinity norm)
- The Mahalanobis distance corrects data for different scales and correlations in the variables
- Inner product space, the angle between two vectors can be used as a distance measure when clustering high dimensional data.
- The Hamming distance measures the minimum number of substitutions required to change one member into another.

Clustering algorithms are generally divided into two categories *Hierarchical clustering* and *Partitional clustering*. Hierarchical clustering build or break hierarchy of clusters. The traditional representation of this hierarchy is a *dendrogram* a tree, when it breaks up then its called *divisive hierarchical clustering* and when it builds then its called *agglomerative hierarchical clustering*. While in partiontinal clustering, dataset is clustering by breaking up into partitions with similar properties.

1.1.3 Components of Clustering Task

According to [JD88, JMF99] main components of clustering task are following:

- Pattern representation or feature extraction and/or feature selection within pattern,
- Definition of pattern proximity measure appropriate to the data domain,
- Clustering or grouping,
- Data abstraction (if needed), and
- assessment of output (if needed).

Pattern representation refers to number of classes, number of available patterns. *Feature selection* is a process of identifying potential subset of feature. *Pattern Proximity* is usually measured by distance function as discussed in previous section. *Grouping* step performs partitioning of data into groups. It usually depends on clustering algorithm. *Data Abstraction* is a process of extraction of a compact representation of dataset, a typical data abstraction is a compact description of each cluster. All clustering algorithm when presented data, clusters data into groups, but how we can determine that clustering algorithm can be characterized into 'good' and 'poor', hence assessment of output is usually desired for evaluation of clustering technique. *Cluster validity* is basically same as cluster assessment with little more 'standards' in determining whether its 'good' or 'poor'.

1.2 Goals

Although WWW is a vast source of information, the problem of *information overload* is more acute than ever. Due to 'noise' in WWW, it is becoming hard to find usable information.

The most immediate cause of information overload on the web is caused by the web is trying to fill the dual role of being both private and public information. Multiple sources and localized content in global environment has increased it further. [Ber97]

The goal of this thesis is to design an IR system which would help to reduce information overload in real-estate industry. Furthermore, using clustering technique it should be able to track real-estate listings.

The importance of an IR system has been greatly increased by advent of the World Wide Web. These IR systems are usually perform *full-text* retrieval systems. However, few IR system combines *full-text*, *data mining* and *artificial intelligence* at rudimentary level to produce outstanding results.

In this thesis we propose an efficient IR system, specially designed for real-estate research industry. Furthermore, we propose a novel clustering technique which detects exact and exact-like² listings called *Weighted clustering*. It is based on *Canopy Clustering* as suggested in [MNU00] and *Locality Sensitive hashing*[GIM99].

The goal of this technique is to examine real-estate listings from multiple sources, identifying exact and near listings and cluster them into similar groups, even if data consist of missing attributes. Currently, real-estate list-

²Exact-like listings are near-neighbours

ings are scattered on multiple portals and often there is a slight difference in listing from two sources. We show through number of tests that proposed technique can be fully apply to real-estate IR system, which improves statistical analysis of real-estates in particular area.

1.3 Assumptions

This thesis is based on the following assumptions:

- In this thesis we use term 'real-time' data but in practice there is a small delay in information from web as it is first crawled by an agent.
- Data used in this thesis was crawled and indexed by a computer program during Jan-2009 till May-2009 from some major real-estate portals in Poland.

1.4 Clusty - An Information Retrieval system

There have been lots of effort in developing real-estate IR system, but most of them do not address two main issues, track real-estate listing on multiple sources and generate insightful trends based on clustering. *Clusty* tries to address these two issue along with other issues. From the functional point of view the goal of this system is twofold:

- Provide a mechanism to crawl information from various sources on WWW.

- Provide a platform for end-users to search, track and generate trends for particular area.

The heart of this system is *inverted index*, which is basically similar to the index present in the end of any textbook. Inverted index stores occurrences of each word in particular listing. Hence, it is searching data become enormously faster. Another component of this system is *Datastore*, which grabs the data from various portals and push on the store server in form of objects. The intelligence unit of this system consists of *Clustering Component*. The statistical unit consists of *Trends Generator*.

1.5 Overview

The structure of this thesis is outlined as following:

Chapter 2 gives an overview of existing research and related work in IR and data clustering, as well as, the background underlying solution proposed in this thesis.

Chapter 3 presents proposed solution in detail, along with key components of this system and algorithms.

Chapter 4 gives a detailed description of design and implementation of the system.

Chapter 5 presents detailed evaluation of proposed system and various tests which confirmed its viability.

Chapter 6 draws the conclusion and possible future extension to this thesis.

Chapter 2

Background and Related Work

In this chapter, we present background behind proposed solution and some of the related work done in past. First, we present various crawling and clustering techniques use in various IR system and then we present related work underlying of current solution present in this thesis.

2.1 Web Crawler

A *Web Crawler* is an automated and methodical computer program which browses the World Wide Web. The process of downloading content from WWW in an automated and methodical way is called *Web Spidering or Crawling*. Web crawlers are mainly used to provide a copy of all visited webpages which can be later processed for research or by search engine to create *inverted index*. Also, crawlers can be used to gather specific types of information from Web pages, for example news articles, real-estate listings, contact cards etc.

2.1.1 Crawling Policies

WWW is not easy to crawl and index, mainly due to large and dynamic nature of WWW. There have been lots of discussing about bandwidth consumed by crawler costs providers.

As [EMT01] stated, "Given that the bandwidth for conducting crawls is neither infinite nor free, it is becoming essential to crawl the Web in not only a scalable, but efficient way, if some reasonable measure of quality or freshness is to be maintained." A crawler must adopt certain policies.

- **Selection Policy:** Given the huge size of WWW, only few percent of webpages have been indexed by various search engines. A web crawler usually select certain urls which are interesting for context in which it is working. For example, Google first crawl upper links, pages that are present in home page and later others. Finding relevant pages is also done by crawlers. In another example, Clusty crawler when it walks the web and detects pages with real-estate listing. This information is feedback by patterns or other parameters. Another interesting method of finding relevancy is *Link Analysis*.
- **Re-visit Policy:** WWW is very dynamic in nature, information appears and disappears more frequently. Hence, a web crawler should have a re-visit policy instead of blind ordinary crawling. We can easily determine *Age* of web page in local copy and if age exceeds t threshold

then re-visit as mentioned below:

$$A_p(t) = \begin{cases} 0 & \text{if } p \text{ is not modified at time } t \\ t - \text{modification time of } p & \text{otherwise} \end{cases} \quad (2.1)$$

However there are two simple re-visit policies based on frequency [CGM03].

In *Uniform Policy* we visit webpage based on uniform frequency and in *Proportional Policy* re-visiting occurs more often the pages that change more frequently. Considering average freshness, the uniform policy is to be applied on URL. If we crawl web page too fast then we are waste both our resouces and website's resources. Hence it is recommended to consider average freshness and construct a crawling policy.

- **Politeness Policy:** Crawlers are automated programs, hence they can crawl information as quickly as possible. The use of Web crawlers is useful for a number of tasks, but comes with a price for the general community. Website bandwidth costs website provider. For example, server load, router crash, network resources and network disruption may cost website host. A partial solution to these problems is the robots exclusion protocol, also known as the robots.txt protocol; It allows website providers to disallow certain bots or agents.
- **Parallelization policy:** Parallelization is use to perform cetain tasks in speedy manner. In case of crawling a parallel crawler is a crawler that runs multiple processes in parallel to crawl webcontent. Idea behind

parallel crawling is to maximize content download rate. Usually in IR system where content is very large parallelization is required. However, parallel crawler have to ensure that it would not fetch requests twice and spam website host.

2.1.2 Crawler Identification

Web crawlers are usually identified by USER-AGENT field in HTTP request. Crawler administrator should use consistent USER-AGENT to allow website administrators to allow or skip crawl. Here is an example from Google Crawler:

Googlebot/2.1 (<http://www.google.com/bot.html>)

2.2 Latent Semantic Analysis

2.2.1 The Classic Vector Space Model

Vector space model or *Term vector model* is a algebraic representation of two text documents as vector of identifiers. It is widely used in IR, indexing and filtering. It was first used in SMART [Sal71] IR system.

The model is based on the following postulates:

- Documents and queries are both vector $\vec{d} = w_{i,1}, w_{i,2} \dots w_{i,n}$ each $w_{i,j}$ is weight representation of term j in i th document. (Bag of words representation)

- Similarity of document vector to a query vector is cosine of angle between them.

$$Sim(d_i, q) = \cos \theta \quad (2.2)$$

The vector space model is divided into three steps:

- Document indexing,
- Weighting of indexed terms,
- Rank the document with respect to the query according to similarity measure.

2.2.2 Latent Semantics Indexing

In IR systems *latent semantics indexing* (LSI) is a general indexing and retrieval method that uses *singular value decomposition* (SVD) to construct relationship between various text in meaningful manner. In basic terms LSI is a principal that two terms are in same context or not. For example consider the following unstructured text:

"Sopot beach and warm sand, pleasures in summer" So, if you ask a LSI based IR system "sand" then it should relate it to "beach".

LSI is rather a computationally expensive [KH00] method. However, modern high-speed processors and inexpensive memory is commonplace now a days, therefore these considerations have been largely overcome. In large scale applications, which basically consists more than millions of document

that need to be processed. On high ended hardware it is not very expensive to computer webgraph.

2.3 Clustering

Clustering problems arises in many different applications, such as data compression and vector quantization [GG92], pattern recognition and classification [DH73], and data mining and knowledge discovery [FPSSU96]. The goal of *clustering* is to reduce the amount of data by grouping them into similar data items. Clustering methods [JD88] are divided into two categories: *hierarchical clustering* and *partitional clustering*. We have discussed about them in previous chapter. In this chapter, we will be discussing various partitional clustering algorithm which evolved the solution proposed in this thesis.

2.3.1 k-means Clustering

k-means clustering is a partitional clustering technique, which aims to partition n datapoints¹ into k clusters in which each observation belong to the cluster with nearest mean [HW79].

Given a set of n data points in real d – *dimensional* space R^d and an integer k , k-means required to determine a set of k points in R^d , called *k-centers*, so as to minimize the mean squared distance from each data point to its nearest center. This measure is called *squared-error distortion* [JMF99,

¹datapoints are dataset elements or in simple term observations

GG92] that is:

$$S = \sum_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - m_i\|^2 \quad (2.3)$$

In general computing k-means is NP-Hard problem [HW79]. There are number of variants to this algorithm, so to clarify we only refer to *Lloyd's algorithm*. It is based on the simple observation that the optimal placement of the center is at centroid of the cluster [DFG99]. The algorithm is divided into two steps as shown in equation below.

$$S_i^{(t)} = \left\{ \mathbf{x}_j : \|\mathbf{x}_j - \mathbf{m}_i^{(t)}\| \leq \|\mathbf{x}_j - \mathbf{m}_{i^*}^{(t)}\| \text{ for all } i^* = 1, \dots, k \right\} \quad (2.4)$$

$$\mathbf{m}_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j \quad (2.5)$$

Equation 2.4 assigns each observation to the cluster with the closest mean. While in equation 2.5 we calculate the new means to be the centroid of the observations in the cluster. Table 1 is showing basic pseudocode of this algorithm.

As we can see k-mean is a heuristic algorithm, hence there is no guarantee that minima is global minima, and the result may depend on the initial selection of clusters. [AV06] has shown that there exist certain point sets on which k-means takes super polynomial time $2^{\Omega(\sqrt{2})}$ to converge, but in practice its hard to observe. This algorithm also suffer from one common problem that minimum is local-minimum.

Below is an example taken from wikipedia, a Voronoi diagram generated

Algorithm 1 K-means algorithm

1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
 2. Assign each object to the group that has the closest centroid.
 3. When all objects have been assigned, recalculate the positions of the K centroids.
 4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.
-

by the means. In step 1 as shown in Figure 2.1 initial "means" (in this case $k = 3$) are randomly selected from the data set (shown in color).

In step 2 as shown in Figure 2.1 clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

In step 3 as show in Figure 2.1, the centroid of each of the k clusters becomes the new means.

In last step as shown in Figure 2.1, the step 2 and 3 has been repeated until convergence has been achieved.

2.3.2 Quality Threshold Clustering

Quality Threshold (QT) is a clustering technique which groups data into clusters ensuring high quality of clusters. Quality is ensured by finding distance between dataset whose diameter does not exceed given threshold. This algorithm is widely used in clustering genetic data which prevents dissimilar

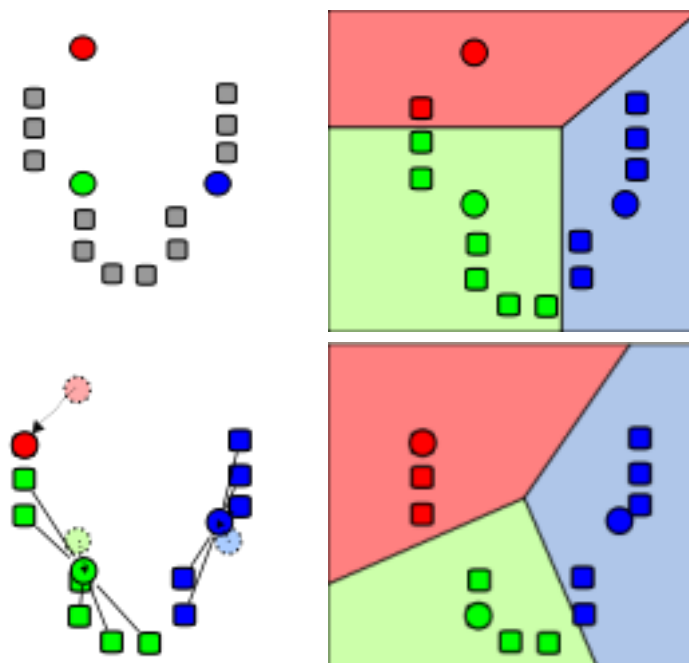


Figure 2.1: K-means illustration

genes cluster in same groups ensuring high quality of cluster [HKY99].

$$d(x, y) < \delta \quad (2.6)$$

In equation 2.6 $d(x, y)$ is distance function and δ is quality threshold. If distance is less than quality threshold then its allowed in cluster else its rejected.

Figure 2.2 depicts graphical illustration of dataset. In this figure one can easily observe the threshold circle.

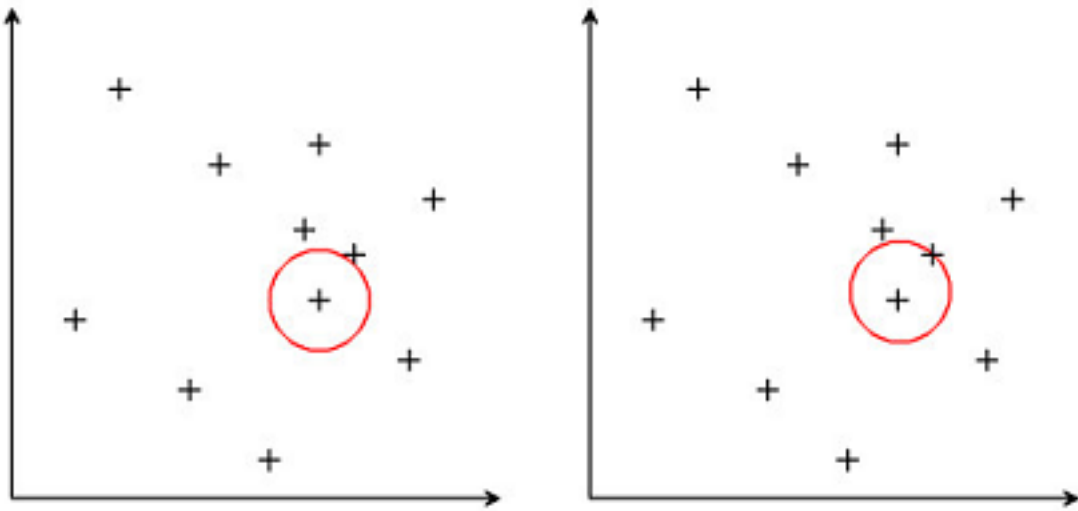


Figure 2.2: QT illustration

Algorithm 2 QT algorithm

1. Select random data points from dataset.
 2. The distance function assign distance between datapoints.
 3. If $d(x, y) < \delta$
Add to cluster
 4. Recurse with the reduced data points.
-

2.3.3 Canopy Clustering

Canopy clustering is a technique proposed by [MNU00] to cluster high dimensional datasets. The key idea is twofold:

- Cheaply partition the data into overlapping subsets, called 'canopies'
- Perform more expensive clustering, but only within these canopies

The key idea of canopy clustering algorithm is one can greatly reduce the number of distance computations required for clustering by first cheaply par-

tion the dataset into overlapping subsets, and then only measure distance between among pair of data in same canopy [MNU00].

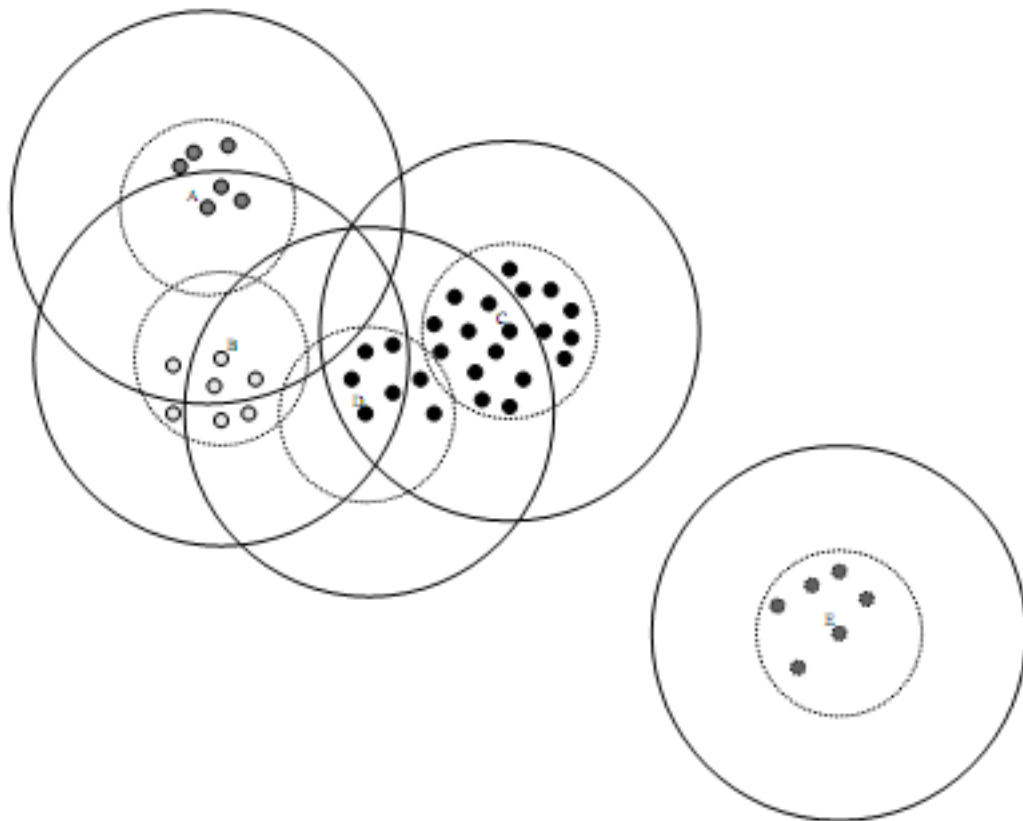


Figure 2.3: An example of canopy clustering: showing four data cluster canopy covering them. C and D are in same canopy, A, B, E are other three canopies.

In figure 2.3 Point A was selected at random forms a canopy consisting of all points within the outer circle. The points inside inner circle are excluded to form canopies. In practice its not hard to create a distance measure for canopy properties.

Algorithm 3 Canopy Clustering Algorithm

1. Select random data points from dataset.
 2. Define two distance T_1 and T_2 such that $T_1 < T_2$ from above datapoint
 3. If $d(x, y) \leq T_1$
Add to cluster and mark it as strong
 4. If $T_1 < d(x, y) \leq T_2$
Add point to cluster and mark it as weak.
 5. Recurse with the reduced data weak data points.
-

Start with a list of datapoints in any order with two distance threshold T_1 and T_2 . These two thresholds are set by user. Pick a point of the list and measure distance between other points. Put all points within T_1 into a canopy and remove all points from list which are within distance threshold T_2 . Repeat until list is empty. Figure 2.3 shows canopies which were created by this method.

The main advantage of this method is one can cluster high dimensional dataset without compromising in quality of clusters [MNU00].

2.3.4 Fuzzy C-Means Clustering

Fuzzy C-Means clustering (FCM) algorithm minimizes the index of quality defined as sum of squared distance for all points included in the cluster space to the center of cluster. In mathematical terms, FCM processes n vectors in p - space as data, and uses them, in conjunction with first order necessary conditions for minimizing the FCM objective functional, to obtain estimates for sets of unknowns. It was first developed by Dunn in 1973 and lately improved by Bezdek in 1981. It based on the minimization of following

objective function:

$$J_m = \sum_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x}_j \in S_i} \mathbf{u}_{ij}^m \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2, \text{ where, } 1 \leq m < \infty \quad (2.7)$$

Here u_{ij} is the degree of membership between i th dimension of d -dimensional of measured data and j th cluster.

Fuzzy partitioning which is carried out through an iterative optimization of the objective function shown above, with the update of membership u_{ij} and the cluster centers u_j by:

$$u_{ij} = \frac{1}{\sum_{\mathbf{N}} \sum_{i=1}^n \frac{\|\mathbf{x}_i - \boldsymbol{\mu}_j\|^{2/m-1}}{\|\mathbf{x}_i - \boldsymbol{\mu}_k\|^{2/m-1}}} \quad (2.8)$$

where

$$\boldsymbol{\mu}_j = \frac{\sum_{\mathbf{S}} \sum_{i=1}^k u_{ij}^m \cdot \mathbf{x}_i}{\sum_{\mathbf{S}} \sum_{i=1}^k u_{ij}^m} \quad (2.9)$$

The termination condition for convergence is when $\max_{ij} \{\|u_{ij}^{k+1} - u_{ij}^k\|\} < \delta$, where δ is a termination criteria.

Algorithm 4 FCM algorithm

1. Select random data points from dataset and mark them as cluster centers.
 2. Assign randomly to each point coefficients for being in the clusters.
 3. while $\max_{ij} \{\|u_{ij}^{k+1} - u_{ij}^k\|\} < \delta$
 4. Compute the centroid for each cluster, using above using (2.9)
 5. For each point, compute its coefficients of being in the clusters, using (2.8).
-

However, FCM suffers from same problem like in case of k-means that is, the minimum is local minimum.

2.3.5 Locality-Sensitive Hashing based clustering

Locality Sensitivity Hashing (LSH) is a probabilistic dimension reduction method of high dimensional data. The basic idea is to hash the input items so that similar items are mapped to the same buckets with high probabilities, that is, the probability of hash collision of similar items is higher.

An LSH family \mathcal{F} is defined for a metric space $\mathcal{M} = (M, d)$, a threshold $R > 0$ and an approximation factor $c > 1$. An LSH family \mathcal{F} is a family of functions $h : \mathcal{M} \rightarrow S$ satisfying the following conditions for any two points $p, q \in \mathcal{M}$, and a function h chosen uniformly at random from \mathcal{F} [GIM99]:

- if $d(p, q) \leq R$, then $h(p) = h(q)$ (p and q collide) with probability at least P_1 ,
- if $d(p, q) \geq cR$, then $h(p) = h(q)$ with probability at most P_2 .

A family is interesting when $P_1 > P_2$. Such a family \mathcal{F} is called (R, cR, P_1, P_2) -sensitive.

Alternatively, an LSH scheme is a family of hash functions H coupled with a probability distribution D over the functions such that a function $h \in H$ chosen according to D satisfies the property that $Pr_{h \in H}[h(a) = h(b)] = \phi(a, b)$ for any $a, b \in U$.

Minhash clustering is based on minHash and LSH family. It is probabilistic in nature and most interesting part is its fast when dataset has higher dimensionality[DDGR07].

÷

	Cheap	Clusters Size	full clustering	High dimensional
K-means	No	Yes	Yes	No
QT	No	No	No	Somewhat
Canopy	Yes	Yes	No	Yes
FCM	Yes	Yes	No	No
LSH based	Yes	Yes	No	Yes

Table 2.1: Comparison of various clustering algorithms

2.4 Real Estate websites

Currently, In Poland most of the real-estate deals are reviewed and evaluated online, that's because of Internet boom. WWW has never been so informative before, real-estate brokers dived into WWW with real-estate portals and that's what motivated us to address the problem of information overload among real-estate portals. We observed during crawling that more than 16% of real-estate listings in Poland exists on multiple portals and around 4% of them exists on all². We've collected data from the following portals:

- <http://gratka.pl>
- <http://szybko.pl>
- <http://trojmiasto.pl>

All these web portals provide basic IR and filtering based on query, some of them geomap³ them and provide a clustering view. But none of them tried to hard cluster them and tried to generate trends in cluster.

²all three portal

³A geo coding mapping on map

2.5 Problems

Most of the problems resides in providing a clear picture to end user. Every portal has its own database. Since, dealers wants attract more consumer, they replication information of major portals which raises issue of *information overload*. Another noteworthy point is there are no automated way to track these listing and *correlate* with other properties. One challenging task in clustering is *data fusion* that is, how to combine information of same objects from various sources. A critical problem which arises that same object on various sources might have missing values. Ideally, In real estate same property should repeat on different sources with same set of data values, but in our crawl we found in 48% of cases same property on different sources is either having missing or different values. It might happen because every portal collect data differently, human mistakes are also considerable in real-world conditions.

Clustering algorithms generally do not handle missing value. Instead, a common solution is *Imputation*, that is, fill in the missing value with appropriate in pre-processing step. However, filled in values are not comparable with observed data, hence data imputation do not produce good results. [TCS⁺01] stated this when evaluation different imputation techniques for biological data: "However, it is important to exercise caution when drawing critical biological conclusions from data that is partially imputed.[...][E]stimated data should beagged where possible [...] to avoid drawing unwarranted conclusions."

Despite above, data imputation is still common practice and imputed data

remain less reliable. In 1994 Ghahramani and Jordan presented modified EM (Expectation-maximization) algorithm that can process missing data. However, this method also estimates maximum like hood model parameters for the missing features. Both Data imputation and EM suffers from the same problem that is, reliability.

2.6 Related work

Data imputation, where missing values are estimated and *marginalization*, where missing values are just ignored are two common method use to handle missing data. In previous section, As we discussed imputation should not be consider as reliable as actual data. We therefore believe that marginalization is better option. Most previous work with marginalization was focused on supervised methods such as Neural Networks [TNA95] or Hidden Markov Model [VGCJ99]. Our approach handles missing data without training data (unsupervised learning).

Considering unsupervised clustering, Fuzzy c-means clustering provides a base to cluster items with missing datum [HB01]. Geometry of missing value in vector is a line or in high dimensionality its a plane, eg. $(2, ?)$ is a line where all x is 2. [HB01] has also suggested following methods:

- Whole Data Strategy, where missing datum is discarded from dataset. In other words its exactly what FCM does. But it does not produce good results.

- Partial distance Strategy, where we calculate partial distance.

$$D_{ij} = \|\mathbf{x}_j - \mathbf{v}_j\|^2 \quad (2.10)$$

$$= \|(1, ?, ?, 3)^T - (2, 3, 5, 3)^T\|^2 = \frac{4}{4-2}((1-2)^2 - (3-3)^2) \quad (2.11)$$

- Optimal completion strategy, where we choose missing value such that we obtain minimal value of J_m where

$$J_m = \sum_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x}_j \in S_i} \mathbf{u}_{ij}^m \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2, \text{ where, } 1 \leq m < \infty \quad (2.12)$$

All above discussed produce good results when less number of features are missing with respect to data dimensionality. In our case we found sometime more than half features are missing.

Also, imputed value lacks reliability when large number of values are missing. We therefore, adopted marginalization as a better solution, which does not create any new data value or estimate best fit. Our tests which are discussed later in this thesis, also indicated that FCM does not fit in web data where data is non structured.

Chapter 3

Algorithms

In this chapter, we present a detailed explanation of proposed algorithm to efficiently cluster web data. First, we described how data look in real-estate IR system and later we described core algorithms which we used in proposed IR system for identifying same object from different sources.

3.1 Feature selection

WWW is non semantic unstructured information. Therefore, mining WWW data has never been easy. Nowadays, WWW is more evolved and information entangled with a real-estate listing or advertisements is more complex. For example, In past, real-estate listings are usually consists of area, price, rooms, floor, street and city, but now information is very much unstructured. In this experiment we collected a variety of data from various sources. Following explain what consists of a datum.

- *Description* string type, generally a blob.

- *Price* float type (required)
- *Area* float type (required)
- *Rooms* float type
- *Floor* float type
- *Total Floor* float type
- *Street* string type
- *District* string type
- *City* string type (required)
- *Geocode* tuple type
- *Type* string type (required)

3.2 Clustering Algorithms

In this work, we used combination of *Canopy clustering* which is a cheaper version of k-means with Fuzzy weighted distance function. We have chosen Canopy clustering because its time-efficient and its quality is nearly same as k-means. First, we divided our features into two set F_s and F_m . F_s is set of features which have to be present in datum while F_m is set of feature which are missing in datum. Marginalization approach would be if we compute clusters using F_s . The idea is to cluster items over F_s considering partial distance over F_m .

3.2.1 Distance function

We used two different distance functions: one without unstructured text and second with unstructured text and during our experimentation we found advantage of combining both unstructured and structured data. In First distance function we computed distance by introducing a relative weight factor w . The normalization of data enables relative importance of some features in datum. For example, while clustering *Type* of property should be same that is, all "Apartments" are clustered together. So, this importance is done via weights $w \in [0, 1]$. As name suggest *Fuzzy* this distance function computes weights dynamically based on fuzzyness¹. Though there are user defined weights to some features, system computes effective weight based on provided weights and *1-norm* distance.

Let us consider weight vector $W = [w_1, w_2 \dots w_i]$ then distance between two real-estate listing can be computed as $d(x, y)$:

$$d(x, y) = \sum_{\mathbf{F}_p \ i=1}^k w_i |x_i - y_i|_1 + \sum_{\mathbf{F}_m \ j=1}^k w_j |x_j - y_j|_1 \quad (3.1)$$

Where x and y are vectors. Since, x_j is missing feature, above can be re-written as:

$$d(x, y) = \sum_{\mathbf{F}_p \ i=1}^k w_i |x_i - y_i| \quad (3.2)$$

In second distance function, we considered the same distance function and combined it with *jaccard distance* considering unstructured text. Jaccard

¹fuzziness is considered in numeric data

distance can be given as:

$$J_\delta(A, B) = 1 - J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} \quad (3.3)$$

The final distance function² looks like

$$D(x, y) = \delta d(x, y) + (1 - \delta)J_\delta(x, y) \quad (3.4)$$

We have chosen $\delta = 0.85$ for our experiments. It means 15% importance is given to unstructured text.

3.2.2 Weighted Clustering

In weighted clustering, we apply canopy clustering to the normalized data by applying relative weights to the feature. During experimentation we have used an variant of this algorithm along with standard algorithm.

Standard weighted clustering algorithm is explained as following:

Algorithm 5 Weighted clustering algorithm

1. Let $C_1..C_k$ be initial canopy centres (randomly choose)
2. For each canopy $C_1..C_k$ lets define two threshold δ_1 and δ_2 also $\delta_1 > \delta_2$.
3. Divide features into F_p and F_m
4. For each point in C_x
5. $d(x, y) = \sum_{\mathbf{F}_p i=1}^k w_i |x_i - y_i|$
if $\delta_2 < d(x, y) < \delta_1$ Mark this point as canopy centre.
6. Repeat Steps 5 until we compute set data in overlapping canopies.

²Jaccard distance is computer only on unstructured text

Above algorithm runs in $O(n \log n)$ and there are some evidences that it performs better than K-means clustering[MNU00].

In an another variant we also examined the same algorithm with individual feature threshold.

Algorithm 6 Weighted clustering algorithm with individual feature threshold

1. Let $C_1..C_k$ be initial canopy centres (randomly choose)
 2. For each canopy $C_1..C_k$ lets define two threshold δ_1 and δ_2 also $\delta_1 > \delta_2$.
 3. Divide features into F_s and F_m
 4. For each point in C_x
 5.
$$d(x, y) = \begin{cases} \infty & \text{if } (x_i - y_i > \eta_i) \text{ feature threshold} \\ \sum_{\mathbb{F}_p, i=1}^k w_i |x_i - y_i| & \text{otherwise} \end{cases}$$
 if $\delta_2 < d(x, y) < \delta_1$ Mark this point as canopy centre.
 6. Repeat Steps 5 until we compute set data in overlapping canopies.
-

Above variant of weighted clustering algorithm is very useful when deviation above threshold is not accepted for clustering into same cluster like in real-estate *floors* and *area*.

3.2.3 Weighted+LSH Clustering

Data collected from web is usually unstructured, For example, description of a real-estate listing is an unstructured text. Computing distance using Jaccard similarity would be tiresome. It is because of N-dimensional data.

LSH performs probabilistic dimension reduction of high dimensional data. The key idea here is to map input data such that similar data will fall in same bucket with high probability. Since number of bucket is much smaller than

dimensionality, LSH provides dimensional reduction.

We combined Weighted Clustering with LSH to utilize unstructured information. As shown in below equation.

$$D(x, y) = \delta d(x, y) + (1 - \delta)J_\delta(x, y) \quad (3.5)$$

Also,

$$Prob[h(x) = h(y)] = J(x, y) \quad (3.6)$$

$$D(x, y) = \delta d(x, y) + (1 - \delta)(1 - Prob[h(x) = h(y)]) \quad (3.7)$$

Clustering algorithm is stated below.

Algorithm 7 Weighted+LSH clustering algorithm

1. Let $C_1..C_k$ be initial canopy centres (randomly choose)
 2. For each canopy $C_1..C_k$ lets define two threshold δ_1 and δ_2 also $\delta_1 > \delta_2$.
 3. Divide features into F_s and F_m
 4. For each point in C_x
 5. $D(x, y) = \delta d(x, y) + (1 - \delta)(1 - Prob[h(x) = h(y)])$
if $\delta_2 < D(x, y) < \delta_1$ Mark this point as canopy centre.
 6. Repeat Steps 5 until we compute set data in overlapping canopies.
-

Above clustering algorithm provides an additional score on unstructured text which might be useful, specially when missing features encountered in the datum. Usually all real-estate sources provide description of a real-estate listing and its highly probable that similar listing will have similar description. For example, if area is missing from datum, but description of this real

estate listing is similar, then LSH provides additional subtraction to distance.

With feature threshold above algorithm can be tweaked to below.

Algorithm 8 Weighted clustering+LSH algorithm with feature thresholds

1. Let $C_1..C_k$ be initial canopy centres (randomly choose)
2. For each canopy $C_1..C_k$ lets define two threshold δ_1 and δ_2 also $\delta_1 > \delta_2$.
3. Divide features into F_s and F_m
4. For each point in C_x
5. $D(x, y) = \begin{cases} \infty & \text{if } (x_i - y_i > \eta_i) \text{ feature threshold} \\ \delta d(x, y) + (1 - \delta)(1 - Prob[h(x) = h(y)]) & \text{otherwise} \end{cases}$
if $\delta_2 < D(x, y) < \delta_1$ Mark this point as canopy centre.
6. Repeat Steps 5 until we compute set data in overlapping canopies.

Again, Above variant of weighted+LSH clustering algorithm is very useful when deviation above threshold is not accepted for clustering.

Chapter 4

Implementation

In previous chapter, we explained the core algorithm behind this thesis. In current chapter, we explain implementation detail of developed system. Most of the component of this system has been newly developed. The purpose of this system it verify and provide "real" picture of harvested data and various clustering methods.

4.1 Clusty - Introduction

Clusty an information retrieval and clustering program is specifically designed to handle real-estate advertisements or listings. In this thesis we focused on "sale" of a real-estate listing. However, we believe clusty should also work to "rental" kind of real-estate listings. Clusty also provide street-level, district-level and city-level trends of real-estate. These trends gives clear image how prices varies in a particular region for similar properties.

Clusty core architecture is based on four building blocks.

- Information Retrieval Unit
- Data Storage Unit
- Data Mining Unit
- Data Rendering Unit

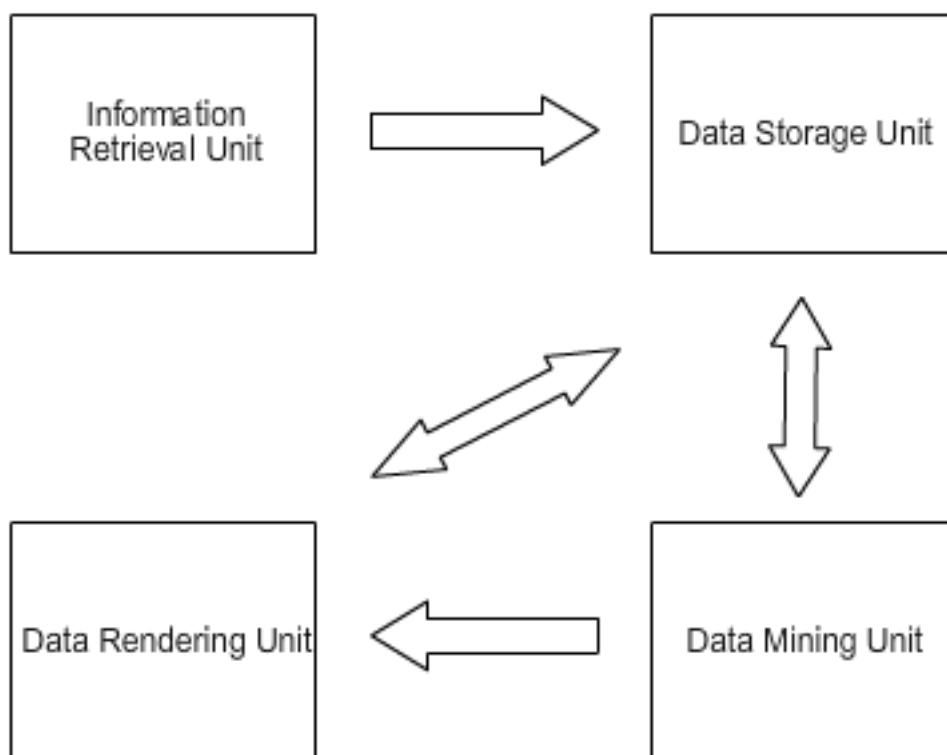


Figure 4.1: Conceptual main units of clusty

Figure 4.1 gives a conceptual view of main units resides into clusty. We will be discussing each unit in detail in this chapter. But, First, we present detailed version of Clusty Architecture.

4.2 General Architecture

Clusty is a platform that consist of four elements: IR unit, data storage unit, data mining unit and data rendering unit. But there are other subsystem units which resides in these four core units. Each unit is responsible for different set of tasks to perform in order to keep system healthy.

Information Retrieval Unit is responsible for talking to World wide web for reading data. A crawling agent EduBot¹ visits pre-defined list of websites or web pages and give back HTML code of these web pages. These HTML is now sent for pattern matching where information is extracted from raw data and then hand it over to Data Storage Unit.

Data Storage Unit is only accessible via Object-Relation Mapper (ORM) to clusty. Information is now serialized into objects and pushed into Database (DB). DB is a relational-schema DB system, in our case we have used MySQL - A free open source RDBMS.

Data Mining Unit, randomly picks unanalysed data from data storage via ORM and analyse it for clustering. Proposed algorithm is being run on data and hand over to DB back.

Data Rendering Unit decided how data should be rendered on User Interface (UI) as per user preferences. Data retrieved from DB is an serializeable object and rendering unit examine various preferences and settings and display it over the UI. Data rendering unit also talks to data mining unit to perform some minor on the fly analysis on data object.

Unlike other existing systems which usually is missing Data mining unit.

¹We use EduBot/1.0 as crawler identifier

Clusty is loosely coupled among these four units and they are kept in modular design so it is very easy to reuse some of existing units to other projects.

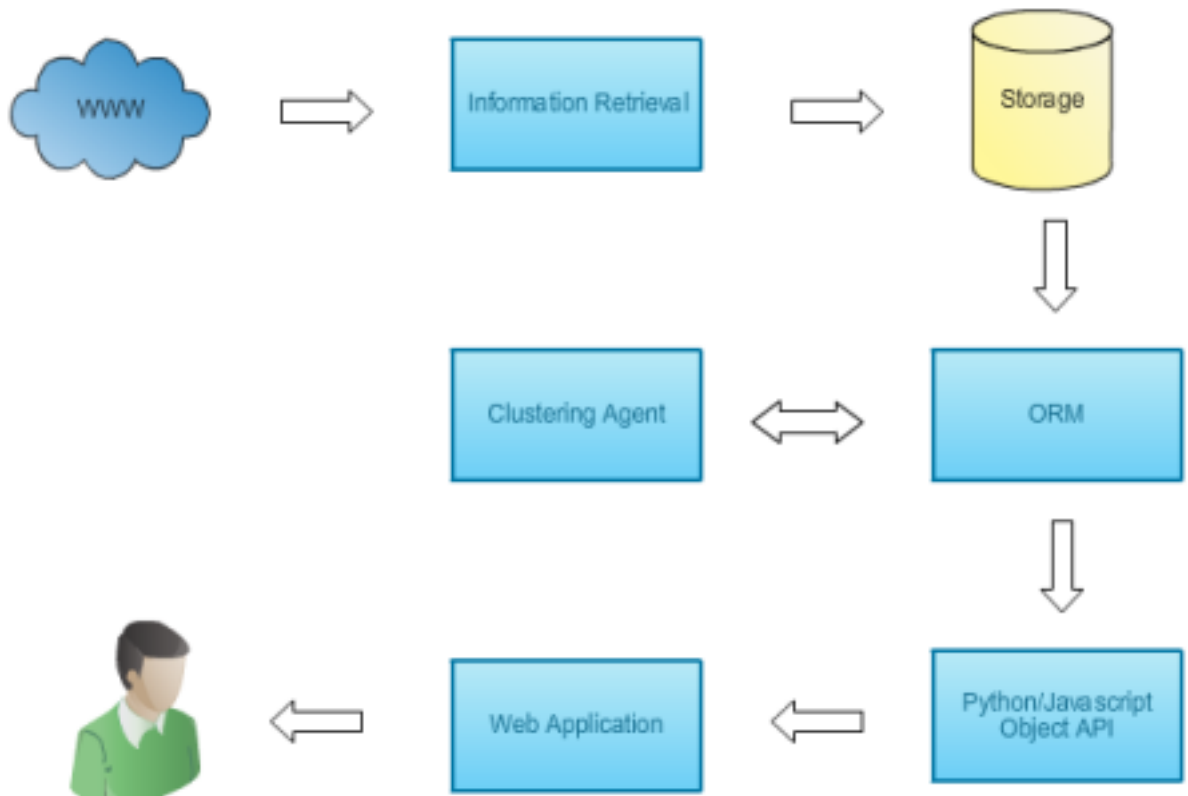


Figure 4.2: Detailed view of clusty units and how they interact

Figure 4.2 shows detailed view of interaction between clusty unit and end user. End user only interact with Web Application UI and other units are invisible to him. However, there is a possibility for an admin user to administrate clusty through a web app, which is specially designed for administration and super users. Database schema is relatively very simple. Figure 4.3 shows clusty database schema. Bold fields are required while gray fields are allowed to be null.

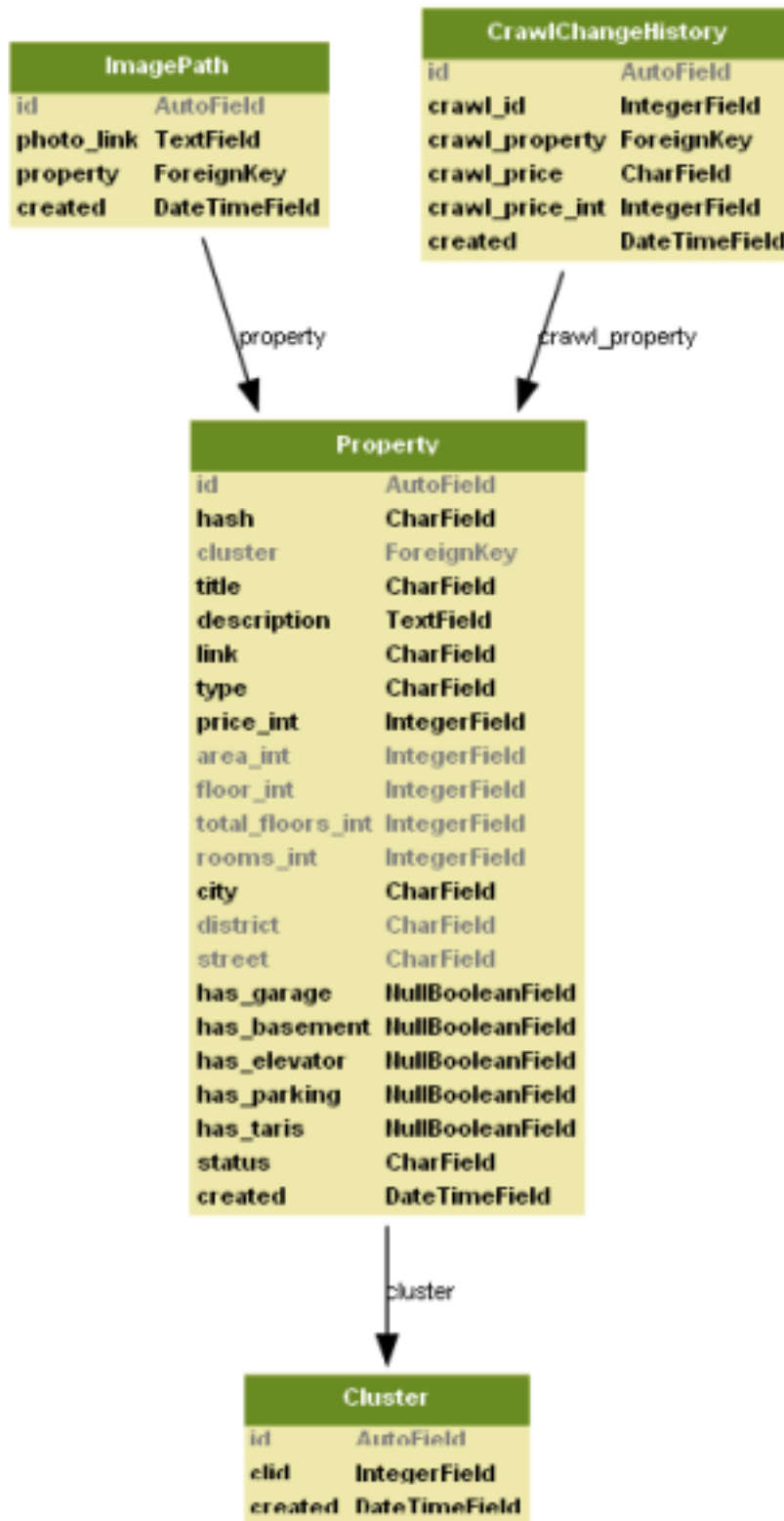


Figure 4.3: Clusty Database Schema

4.3 Development Environment

Clusty is purely implemented in Python Programming Language. Some external components for verification of algorithms are implemented in Matlab. Since Clusty is written in Python, which is a platform independent interpreted language, clusty runs on all existing OS (windows, Linux and MacOS based) and its UI is web based it is reachable from any Web Browser.

Below are some development facts about Clusty

- It runs on Windows, Linux and MacOS without Virtual Machine
- Web UI support IE6+, Firefox, Opera and Safari.
- Written in Python2.5 and some development components are gracefully taken from Django Project.
- Implemented ORM supports SQLite, MySQL, PostgreSQL and Oracle.
- Rendering engine supports i18n (internationalization)

4.4 System Overview

The design of the system can be split into the following components.

- Crawling Agents (IR unit)
- Text Processing Component (IR unit)
- Document Repository (Storage unit)
- ORM and Query Analyser Component (Storage unit)

- Clustering Component (Mining unit)
- Context Renderer Component (Rendering unit)

All of the above components are implemented purely in Python Programming Language as part of this work.

4.4.1 Crawling Agents

Crawling agents are responsible for fetching required pages from WWW. In principal, they visit TODO list and pick one task, in our case its url to be fetched. Like other existing crawling agents they also share their name with host and if host crawling policy allows then it creates socket connection and read content.

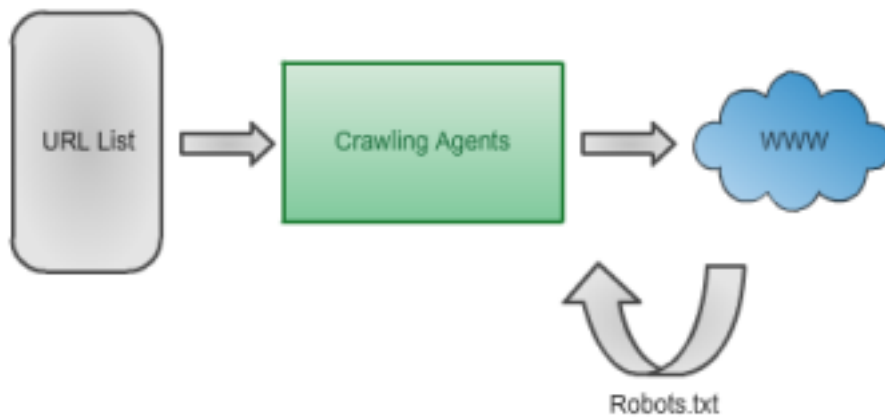


Figure 4.4: Detailed view of Crawling agents

In generic sense, crawler first talk to host and asks for "robots.txt" crawling agent read this file and see if he has got permission to read other urls on providers host. If permitted crawling agent proceed further and request for

an URL. Figure 4.4 shows detailed view how crawling agents interact with WWW and the system.

Another aspect of is to appending URL list by seeing links in crawled document. Usually Crawling components are doing this, but in clusty we implemented it in processing component. The reason for this are:

- Different sources use different URLs. It has been observed that URLs which led to real-estate listing is different on different sources some uses. Some uses `/ref/12345` other uses `/mieszkania-2-pok-warszawa`, for sake of normalization we kept it in text processing component.
- Reducing workload from connection based module. Since crawling agents are connection oriented agents, we kept all kind of text processing out of them to work faster.

Crawler agents are independent and runs in thread to work in parallel. There is safe thread check between agents to avoid re-doing of same task.

4.4.2 Text Processing Component

As name suggests, text processing component processes raw text from WWW to extract useful information from it. In clusty, it is a part of IR unit and coupled between crawling agents and storage.

Individual source has its own text processing component. For example, there is an individual set of patterns to be matched against text for different sources. These patterns resides into common Regular Expression (RegEx) Engine.

Key Component of Text processing are:

- HTML Processor
- RegEx Engine
- Information Extraction
- URL Extractor

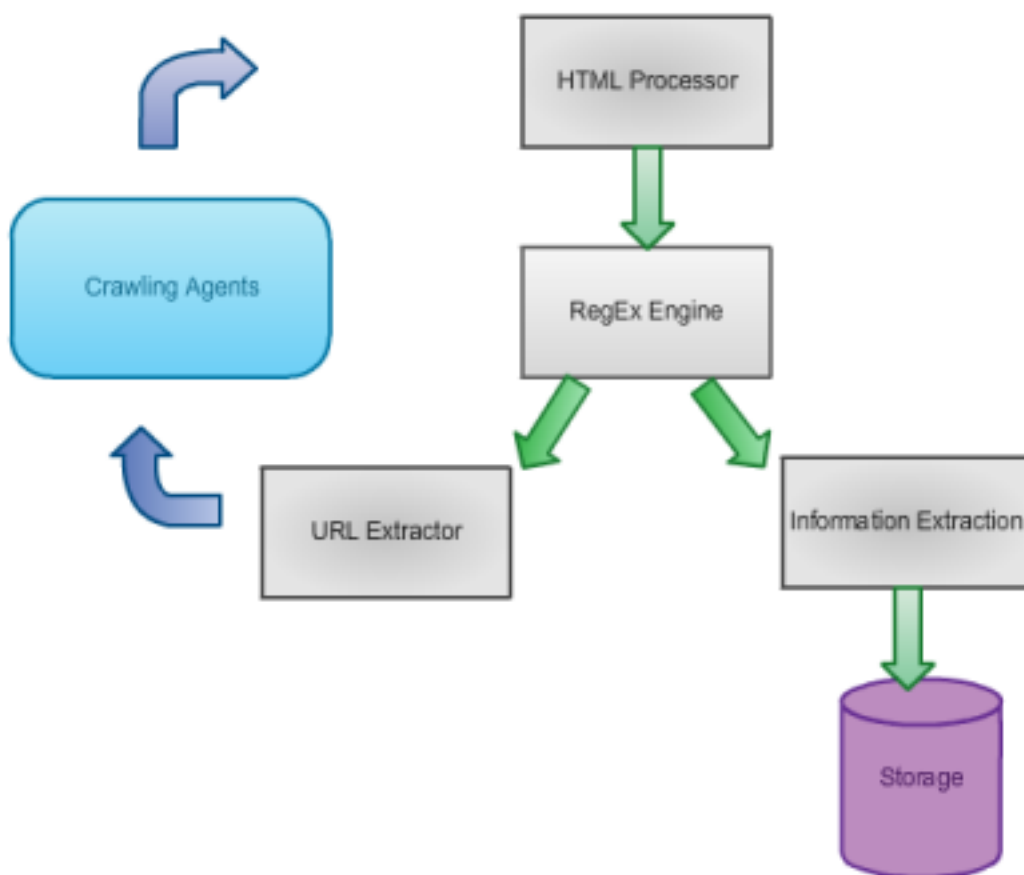


Figure 4.5: Detailed view of Text Processing component

Figure 4.5 shows how key components of text processing interact with each other. Crawling agents pass HTML code to the HTML processor, key

role of HTML Processor is to strip down HTML text into a structured DOM. Searching pattern in whole HTML could be tiresome and time consuming to keep it robust, we decided to construct a parsed Data structure.

Parsed HTML is now chunked into smaller sections of parsed HTML to be matched against pattern. Tricky part of implementation is to have a RegEx Engine that understand different portal and different HTML type. Clusty use Python re module (which is regular expression equivalent in Java) we extended this module and wrote a Generic pattern matcher.

Each source has its own regular expressions or parsing grammar. These grammars or regex are available to RegEx engine dynamically when crawler agents embed source in HTML. Crawler agent tells HTML processor for which source it have got the HTML. This important information is now passed to RegEx Engine to load some of its component and lately the same information is passed to URL extractor and information extraction module, where crucial information is being taken out from parsed data.

In our experiment we have found that generally sources site remains polite they allow crawler to index web pages and oftenly they do not make big structural difference.

Information is now in an encrypted storage system.

4.4.3 Document Repository

Document repository is where all crawled and processed documents are stored. It is basically a storage system with non-conventional assumptions. Since, information is needed to be recall or search by queries. Document reposi-

tory is designed to keep Multiple Version and it should have Multi version concurrency control (MVCC). Also, an inverted index is constructed to ease search and query operations on the Document Repository. We are stripping all HTML tags and extracting text and this text is further used to construct inverted index.



Figure 4.6: Detailed view of Data repository

Figure 4.6 shows detailed design of document repository. In center is storage system which is RDBMS storage system with inverted indexer and Multi version concurrency controller.

Document arrives at inverted index. It get splitted into array of words and then we strip all common words². Lets consider an example $D_0 =$ "My name is Alice" and $D_1 =$ "My name is Alice and my best friend is Bob".

The we create an inverted index for these two documents, assuming {"a", "is", "my", "and"} are stop words.

Table 4.1 shows how inverted index is stored. The main advantage of inverted index is that we can query inverted index and lookup time is dra-

²We process all data for Polish language

word	document
name	{0, 1}
Alice	{0, 1}
best	{1}
friend	{1}
Bob	{1}

Table 4.1: An Example of an inverted index

matically smaller. For example if $q = \text{bob and alice}$ it means $\{1\} \cap \{0, 1\}$ which is 1. In simple words, Document 1 contains above query.

Clusty uses very simple implementation of MVCC: timestamp+md5 based control checking. Clusty request a document to WWW it arrives with timestamp if document timestamp and md5 check sum matches with existing document then its discarded else it is revisioned incrementally.

4.4.4 ORM and Query Analyzer

Object-relational mapper (ORM) is a technique to map data between incompatible type systems in relational database system and object oriented (OO) like programming language. We can directly store objects in database table using ORM. For example, consider an object *Person* below:

```
class Person(object):
    first_name = charField()
    last_name = charField()
    address = addressField()
    email = emailField()
```

is mapped to a table in database

```
CREATE TABLE person (  
    first_name char(50),  
    last_name char(50),  
    address char(50),  
    email char(50),  
)
```

Since objects are apprehensible in programming language while SQL uses different route, it is always preferred to use ORM for neat implementation. Another advantage is code is plug able to many RDBMS. For example, clusty code can be run with MySQL, PostgreSQL, sqlite and Oracle.

Beside many advantages there are some disadvantages also, it is hard to get it working with non SQL RDBMS (schemaless DB) and the are not very well optimized for big tasks. Although, there are some disadvantages we still used ORM model because our data model is very simple and it does not need high ended SQL.

Figure 4.7 shows how ORM and Query Analyzer interact with other system. Query Analyzer is built inside ORM to understand some of very common searchable queries. This work is not focused on Natural Language Processing (NLP), so we did not tried to develop any analyser which can translate natural language into queries. But, to make query and filtering easy we developed a *Key:Value* based analyser which also has some of NLP like features.

Table 4.2 is showing some of example queries and how it is apprehend by the system. Queries are parsed by a simple parser which extract key, value

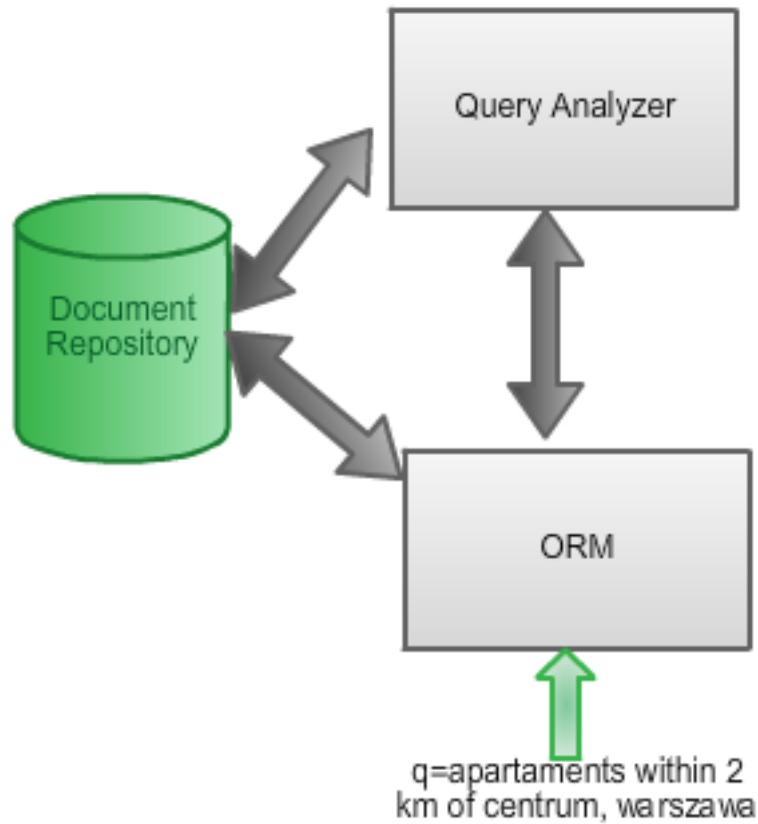


Figure 4.7: Detailed view of ORM and query component

from it and ask database to call SQL query. It gives flexibility of querying any column by common sense instead of knowing high ended SQL language. That is why query analyser resides in ORM.

4.4.5 Clustering Component

Clustering component is a data mining component which is responsible for clustering unclustered data. Previous chapter, gives detailed idea about clustering algorithm used in the system. This component has implemented LSH,

Human query	Machine query
apartment in Warsaw	select all listing where type = apartment and city = Warsaw
land within 2km of centrum, Warsaw	select all listings where type = land and distance = 2 loc = centrum, warsaw
apartments maxprice:200000	select all listings where type = apartments and price = 200000
apartments since:"2 weeks"	select all listings where type = apartments and timerange = 2 weeks
apartments warsaw or gdansk	select all listings where type = apartments and city warsaw or city = gdansk

Table 4.2: An Example of how queries are translated

Weighted and Weighted+LSH algorithms.

Clustering component has two main modules, which ensure data clustering integrity and clustering logic:

- Random Data Selector
- Clustering Logic

Figure 4.8 shows basic module of clustering component. Random data selector selects data from ORM, that need to be clustered or analysed and after being analysed it mark item as clustered. Randomly picked data now clustered in global scope by clustering logic module and send back to ORM which push back to Database. Clustering component can be invoked by administrator and can be run on a cron job basis. We use cron job approach in this work.

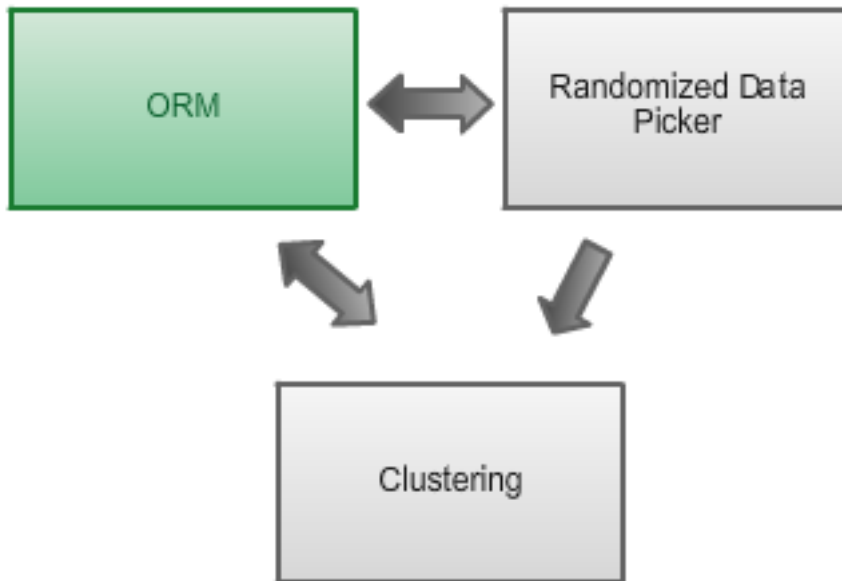


Figure 4.8: Detailed view of clustering component

It is also interesting to understand that what happen during clustering. When an unclustered listing appears its distance is measured from center of existing clusters and run as per algorithm. However, we continue to expand existing cluster if algorithm permits otherwise we create a new cluster. We do not split existing clusters in clusty. Also, We purge automatically records later than 6-month for clustering purpose, it means records are available in database for reference but we do not consider them for clustering purpose.

4.4.6 Context Rendering Component

Context rendering component is a serializing unit which serializes ORM data objects into JavaScript Object Notion (JSON). Main advantage of having Context renderer that we can extract context variables from templates and

replace them with actual data. Another important thing is context renderer returns common format, which can be reusable on different websites just like a web Application Program Interface (API).

For example. A template may look like:

```
Appartament - { obj.rooms } rooms - { obj.area }sqm area
```

and JSON may look like:

```
{"obj":{"area":"50", "rooms":"3"}}
```

Context renderer renders combines both and render them as

```
Appartament - 3 rooms - 50sqm area
```

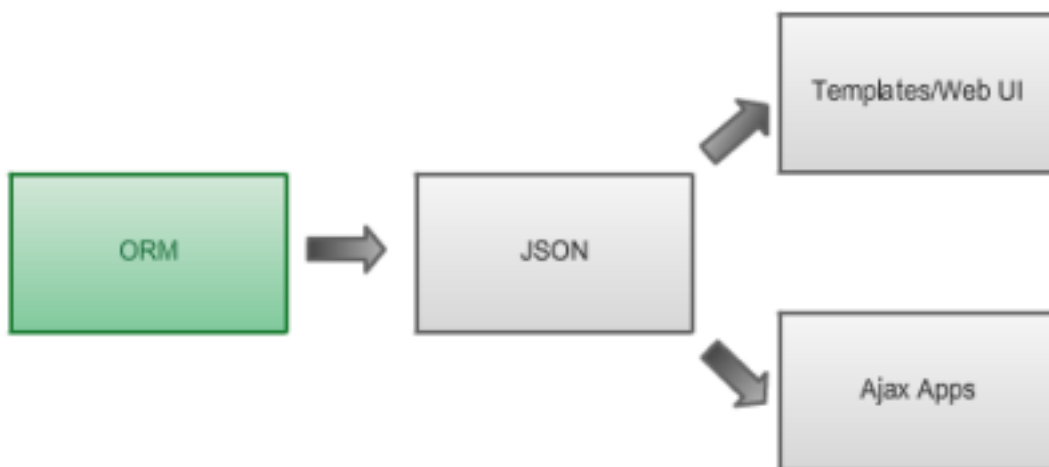


Figure 4.9: Detailed view of context rendering component

Figure 4.9 explains above ideal visually. Advantage of JSON middle ware is that we can create many application which utilize same data without talking to ORM. In this work we have create two applications:

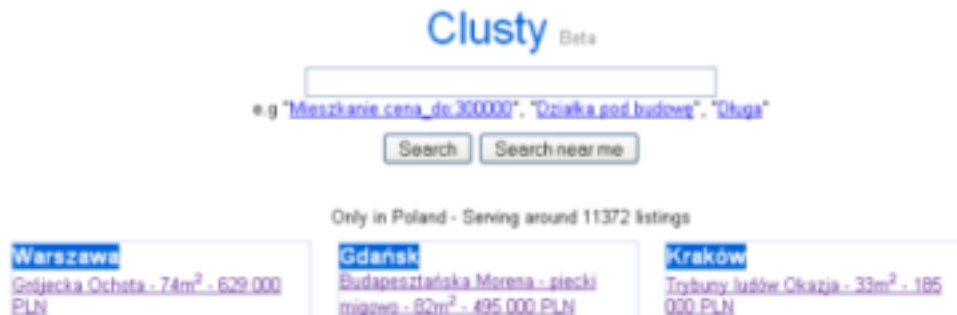


Figure 4.10: Main Screen

- Web App
- Admin App

They both talk to same JSON component for data and variable resolution.

4.5 Application User Interface

Clusty interface is divided into two parts (1) General User Interface (2) Admin interface. In this section, we will be mainly discussing general user interface.

4.5.1 Main Screen

Figure 4.10 shows Main query screen. Just like any other searching system it accepts human queries, populate three most active cities in Poland (real-

estate listings). It also allows to set you localization which is helpful in getting queries like $q = \text{apartments within 2km}$

Main screen also pre-populates some of most recent listings and indicates basic statistics about number of real-estate listing in the system. There are some basic example queries.

Main screen is main entry point to application. It is very intuitively and based on other existing system.

4.5.2 Result Screen

Result screen displays results which returned by query in list or tabular format. Figure 4.11 visually shows how results are rendering in clusty.

Results are rendered in a list instead of conventional tabular format but this list has all possible table operation including column sort.

4.5.3 Filter Screen


Filter screen allows filtering of existing results return by the query. Filter actually constructs a new query keep in mind of what originally was asked and request new query to ORM while user feels it actually filtered results.

Figure 4.12 shows filter screen visually. The reason why it fires a new query is keep query standard. Since query analyser support key:value like structure user can query on columns and numeric ranges. This feature is very useful in filtering.

Clusty

Sort by: Showing 1 - 10 of around 1 052 results for **Mieszkanie cena_do:3**

[mieszkanie 2-pokojowe, 55 m², parter](#)

 Dąb, Katowice
192 000 PLN 55m² 2 Rooms
1 week, 2 days ago
<http://dom.gratka.pl/tresc/397-27252785-slaskie-katowice-dab.html>

[Gdańsk, ul koralowa, 3-pokoje](#)


 Koralowa, Orunia góra, Gdańsk
255 000 PLN 52m² 3 Rooms 4th floor 10 Floors in total
1 week, 2 days ago
<http://ogloszenia.trojmiasto.pl/nieruchomosci-sprzedam/gdansk-ul-koralowa-3-pokoje-og1642030.html>

Figure 4.11: Results Screen

Filter results

Price
 to PLN

Area
 to m²

Rooms
 to

Floors
 to

Total Floors
 to

Within
 km from

Figure 4.12: Filter Screen

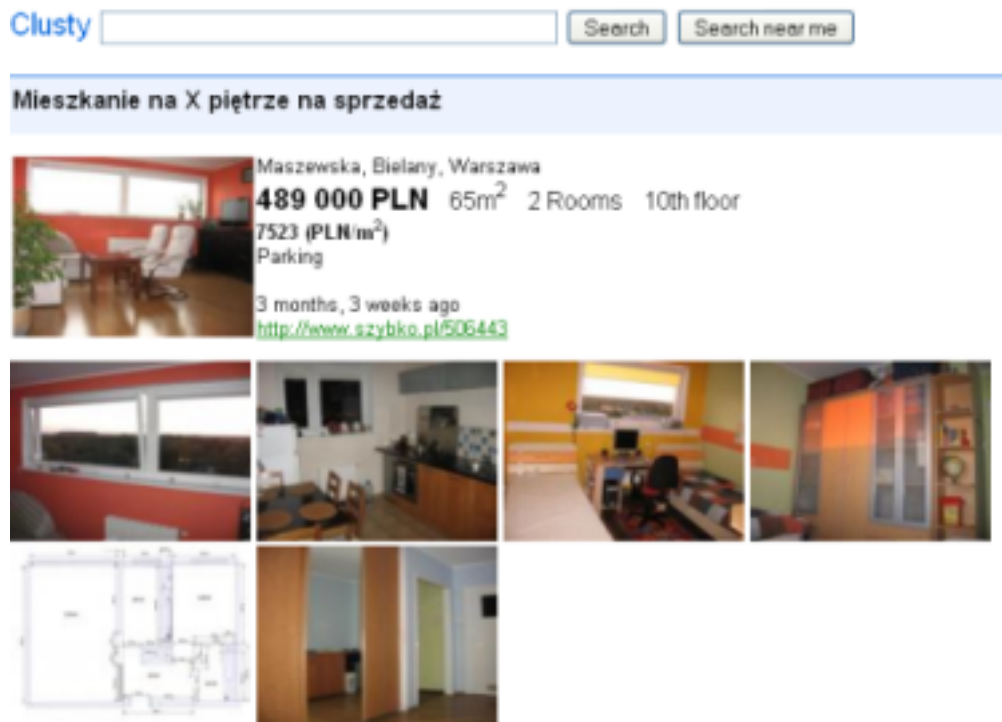


Figure 4.13: Detail Screen

4.5.4 Detail Screen

If visited, then result links takes user to detailed information list. Which gives detailed information with geolocation on map. Along with how listing on current street, district and city is changing with respect to time. Also, it shows properties in same cluster.

Figure 4.13 shows detailed information of listing with pictures.

Figure 4.14 shows clustered screen.



[Mieszkanie na parterze na sprzedaż](#)
Plastowska, Przymorze, Gdańsk - 55m² - 447 000 PLN

[Mieszkanie na parterze na sprzedaż](#)
Plastowska, Przymorze, Gdańsk - 55m² - 439 000 PLN

[Mieszkanie na parterze na sprzedaż](#)
Plastowska, Przymorze, Gdańsk - 55m² - 440 000 PLN

[mieszkanie 3-pokojowe w Oliwie w nowym budownictwie](#)
Plastowska, Przymorze, Gdańsk - 55m² - 439 450 PLN

Figure 4.14: Clustered item on detail screen

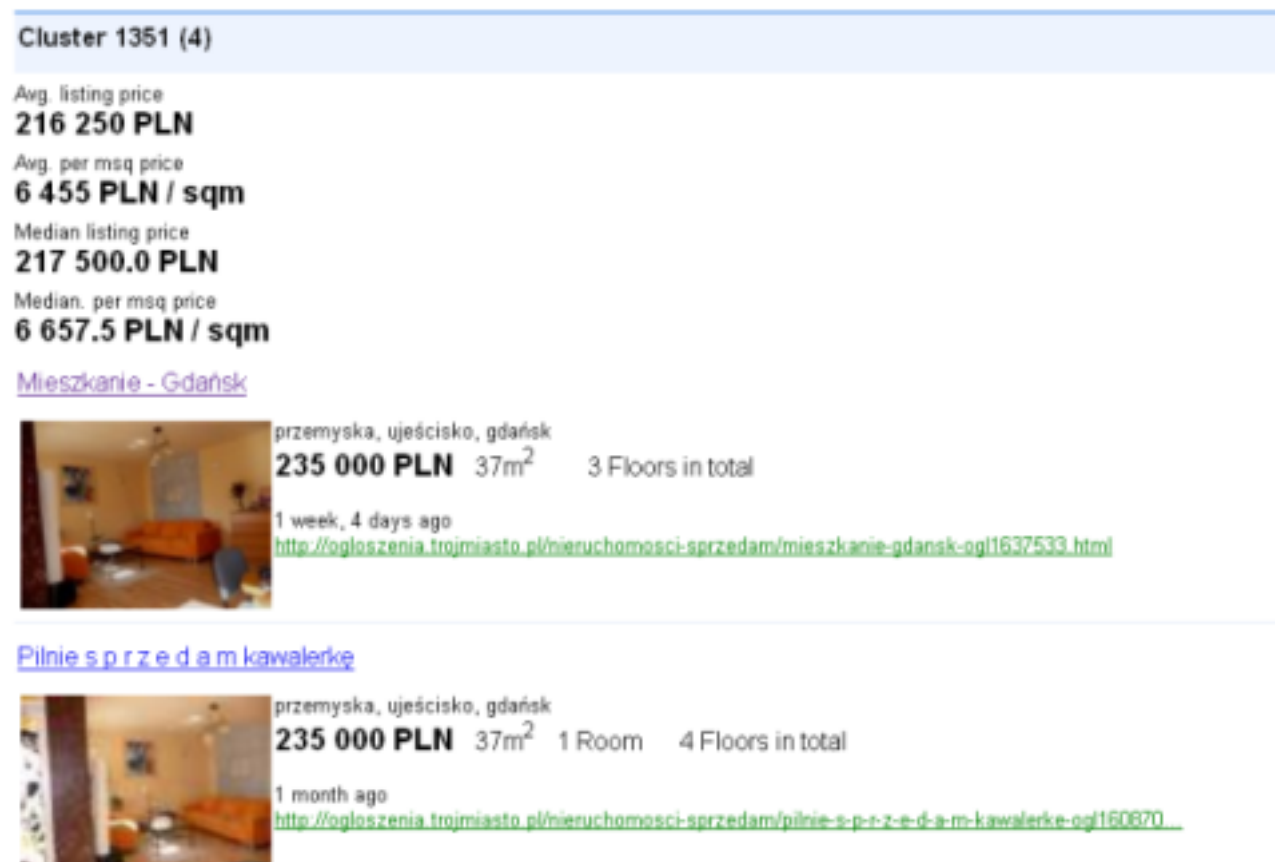


Figure 4.15: Clustered screen

4.5.5 Cluster Screen

Figure 4.15 shows cluster screen which cluster statistics. Screenshot is self explanatory.

4.5.6 Trend Screen

Trend generator dynamically creates trend for items. For example, if there is a listing then trend generator searches items in same city, district and street

Avg. listing price
534 833 PLN -4567 PLN
 (-0.85 %) (w-o-w)
 Avg. per msq price
7 843 PLN -388 PLN (-4.71
 %) (m-o-m)



Figure 4.16: Trend screen

to display trend to user.

Figure 4.16 shows Trend screen prices with respect to time for a city. Screenshot is self explanatory. Trends are generated by the system on the fly. System detects which trends it being asked for and it computes trend and keep it in cache memory for 1 hour.

Home > Portal > Cluster Settings > recommended cluster setting parameters

Change Cluster Setting

Setting name:	<input type="text" value="recommended"/>
	<small>Must be unique</small>
Diff in price in %:	<input type="text" value="15"/>
Diff in area:	<input type="text" value="5"/>
Diff in floor:	<input type="text" value="2"/>
Diff in rooms:	<input type="text" value="2"/>
<input checked="" type="checkbox"/> Don't cluster items if they go beyond difference	

Figure 4.17: Clustering parameters UI

4.5.7 Admin parameter settings

Admin can adjust clustering parameter to be strict on some features or to allow flexibility if needed. Figure 4.17 shows basic UI of setting panel.

4.5.8 Admin Management Screen

Figure 4.18 shows admin console, which is a very useful tool for admin to manage and configure program and clustering parameters. Admin panel allows to define weights on features for example, administrator may set weights for price, area, floors, rooms and admin can also define allowed range for example, price difference can be 15K and area difference can be 2sqm etc.

Home / Portal / Properties

Select property to change Add property

Q | Search

Actions: ----- Go

<input type="checkbox"/> Title	Price int	Area int	Rooms int	Floor int	Total floors int	Street	City
<input type="checkbox"/> Mieszkanie 1-pokojowe, 38 ^{m²} , 4 piętro	125000	38	1	4	None	1000-lacie	Katowice
<input type="checkbox"/> Mieszkanie 1-pokojowe, 33 ^{m²} , 4 piętro	105000	33	1	4	None	Trybuny Ludów	Kraków
<input type="checkbox"/> Mieszkanie 3-pokojowe, 74 ^{m²} , 12 piętro	629000	74	3	12	0	grójecka	warszawa
<input type="checkbox"/> Mieszkanie 2-pokojowe, 55 ^{m²} , 2 piętro	528000	55	2	2	0	Jana Kazimierza	warszawa
<input type="checkbox"/> Mieszkanie 3-pokojowe, 80 ^{m²} , 3 piętro	650000	80	3	3	0		katowice
<input type="checkbox"/> Mieszkanie 3-pokojowe, 54 ^{m²} , 1 piętro	550000	54	3	1	None	Św. Jacka	KRAKÓW

Filter

By created

- Any date
- Today
- Past 7 days
- This month
- This year

By type

- All
- Biuro
- dom
- dom bliźniak
- Dom kamienica
- dom piętrowy
- Dom rekreacyjny
- Dom szeregowy
- dom wolnostojący
- Dziśka
- Dziśka handlowa
- Dziśka ogrodnicza
- Dziśka pod budowę
- Dziśka przemysłowa

Figure 4.18: Admin management screen

Chapter 5

Evaluation

This chapter is dedicated to evaluation of proposed technique in this work. First, we present program testing and testing environment itself, which is later followed by evaluation of proposed algorithm also a brief comparison with some of existing techniques and algorithms. In this chapter, we focused on detailed evaluation of clustering algorithm instead application itself or IR evaluation.

5.1 Unit Testing

Unit testing is a software testing and verification mechanism, which ensures that individual unit of program are fit for use. We have wrote 33 Unit tests to ensure validity of our build and to make sure every component is working as expected. An example output from buildbot is shown below. Basically, it creates test database loads some test data and then ran tests on application. Around 33 unit tests runs on Clusty which validates working of

each component of Clusty.

```
Creating test database ...
Creating table admin_log
Creating table auth_permission
Creating table auth_group
Creating table auth_user
Creating table auth_message
Creating table portal_cluster
Creating table portal_property
Creating table portal_imagepath
Creating table portal_crawlchangehistory
Installing index for admin.LogEntry model
Installing index for auth.Permission model
Installing index for auth.Message model
Installing index for portal.Property model
Installing index for portal.ImagePath model
Installing index for portal.CrawlChangeHistory model
Installing json fixture 'initial_data' from absolute path.
Installed 1001 object(s) from 1 fixture(s)
.....
Ran 33 tests in 3.635s
```

OK

Destroying test database...

5.2 Testing Environment

We have created a random dataset from crawled data for evaluation of proposed technique. In this dataset, We have 1001 objects at random from database and we run clustering algorithm in virtual environment and we used same environment of comparing all different algorithms.

We will compare following existing techniques with (a) Weighted Clustering (b) Weighted+LSH clustering

- k-means clustering
- minhash (LSH) Clustering
- FCM clustering
- FCM+LSH clustering

We investigate all of above in twofold

- Exact Matching Problem (EMP)
- Near-Neighbour Matching Problem (NNMP)

In EMP we examine exact real-estate listing with exact or missing features while in NNMP we examine near listing based on user threshold and it may also have missing features.

We measure *Purity* which is basically transparency evolution measure. Formally it is given by

$$purity(S, C) = \frac{1}{N} \sum_{\mathbf{k}} \max |s_k \cap c_k| \quad (5.1)$$

where $S = \{s_1..s_k\}$ is set of cluster and $C = \{c_1..c_k\}$ is set of classes.

We also measure *Rand Index* which is given by below:

$$RI = \frac{TP + TN}{TP + TN + FP + FN} \quad (5.2)$$

Where, TP is True Positive, TN is True Negative, FP is False Positive and FN is False Negative.

5.3 Reference classes

We first cluster dataset manually to verify the validity of mentioned algorithm. This classification is done manually by data analysis and inspecting images by human. We have formed 168 cluster for EMP and 192 for NNMP.

5.4 Results

5.4.1 k-means

We first analyzed how k-means perform on selected dataset. k-means is one of the famous and easiest clustering technique.

From Table 5.3 we drived that k-means is very time consuming, however cluster size is considerable. Results shows that k-means clustering performed

	Same cluster	Different clusters
Same class	247	68
Different class	31	144

Table 5.1: k-mean EMP Contingency table

	Same cluster	Different clusters
Same class	269	166
Different class	131	217

Table 5.2: k-mean NNMP Contingency table

with 80% accuracy in case of EMP and 66% in case of NNMP.

Type	Time (Sec)	Avg Cluster Size	Purity	Rand Index
k-means EMP	4.2	5.4	0.84	0.80
k-means NNMP	5.6	6.6	0.72	0.66

Table 5.3: k-mean performance on dataset

5.4.2 FCM

	Same cluster	Different clusters
Same class	265	42
Different class	33	229

Table 5.4: FCM EMP Contingency table

	Same cluster	Different clusters
Same class	341	148
Different class	162	301

Table 5.5: FCM NNMP Contingency table

Type	Time (Sec)	Avg Cluster Size	Purity	Rand Index
FCM EMP	3.1	4.2	0.88	0.87
FCM NNMP	4.8	5.4	0.70	0.66

Table 5.6: FCM performance on dataset

From Table 5.6 clustering accuracy is higher in EMP case while incase of NNMP its performing very similar to k-means with accuracy of 66%.

5.4.3 minHash (LSH)

From Table 5.9 clustering accuracy mediocre in EMP while incase of NNMP its performing very low 60%. Noteworthy point is minHash (LSH) based clustering runs very quickly on data. Though we believe our implementation was not optimized for time.

	Same cluster	Different clusters
Same class	221	61
Different class	39	204

Table 5.7: LSH EMP Contingency table

	Same cluster	Different clusters
Same class	280	197
Different class	189	298

Table 5.8: LSH NNMP Contingency table

Type	Time (Sec)	Avg Cluster Size	Purity	Rand Index
MinHash EMP	1.2	7.2	0.83	0.81
MinHash NNMP	1.4	8.4	0.71	0.60

Table 5.9: LSH performance on dataset

Table 5.9 shows LSH does not perform very well in both cases, the reason for this is information is structured and missing or alter values make it worse for pure LSH based algorithm.

5.4.4 Weighted clustering

Now, we test proposed technique. We believe Weighted clustering is remedy of structured information where some attributes should be given more importance than other. Table 5.12 supports our claim. Since data is hybrid often in case of RDBMS driven applications, algorithm focus on high weights features and do no impute missing features which leds to good clustering.

	Same cluster	Different clusters
Same class	289	11
Different class	8	233

Table 5.10: Weighted EMP Contingency table

	Same cluster	Different clusters
Same class	366	77
Different class	63	350

Table 5.11: Wighted NNMP Contingency table

Type	Time (Sec)	Avg Cluster Size	Purity	Rand Index
Weighted EMP	1.6	5.2	0.98	0.97
Weighted NNMP	1.9	5.8	0.89	0.84

Table 5.12: Weighted clustering performance on dataset

Clustering algorithm run very quickly and able to detect exact replica while some features are missing with high precision.

5.4.5 FCM+LSH

Now, we utilize unstructured information in this technique. Idea is to combine FCM score with LSH score only over unstructured data. Table 5.15 shows that results are much better than standalone FCM.

	Same cluster	Different clusters
Same class	276	41
Different class	30	238

Table 5.13: FCM+LSH EMP Contingency table

	Same cluster	Different clusters
Same class	355	129
Different class	111	303

Table 5.14: FCM+LSH NNMP Contingency table

Type	Time (Sec)	Avg Cluster Size	Purity	Rand Index
FCM+LSH EMP	2.4	5.2	0.93	0.89
FCM+LSH NNMP	3.2	5.7	0.75	0.73

Table 5.15: FCM+LSH performance on dataset

5.4.6 Weighted+LSH

We combine LSH with Weighted algorithm as suggested in chapter 3. Experiment using this algorithm show very fine accuracy in EMP and acceptable NNMP. Table 5.18 shows that accuracy is around 90% for NNMP and 98% for EMP.

	Same cluster	Different clusters
Same class	303	7
Different class	4	237

Table 5.16: Weighted+LSH EMP Contingency table

	Same cluster	Different clusters
Same class	392	49
Different class	35	398

Table 5.17: Weighted+LSH NNMP Contingency table

Type	Time (Sec)	Avg Cluster Size	Purity	Rand Index
Weighted+LSH EMP	2.0	5.3	0.98	0.98
Weighted+LSH NNMP	2.2	5.8	0.93	0.90

Table 5.18: Weighted+LSH performance on dataset

5.5 Comparison Plot

Figure 5.1 Accuracy for all six algorithms for both model (exact and near). Combination of Weighted and LSH performs better in both models than other compared algorithms.

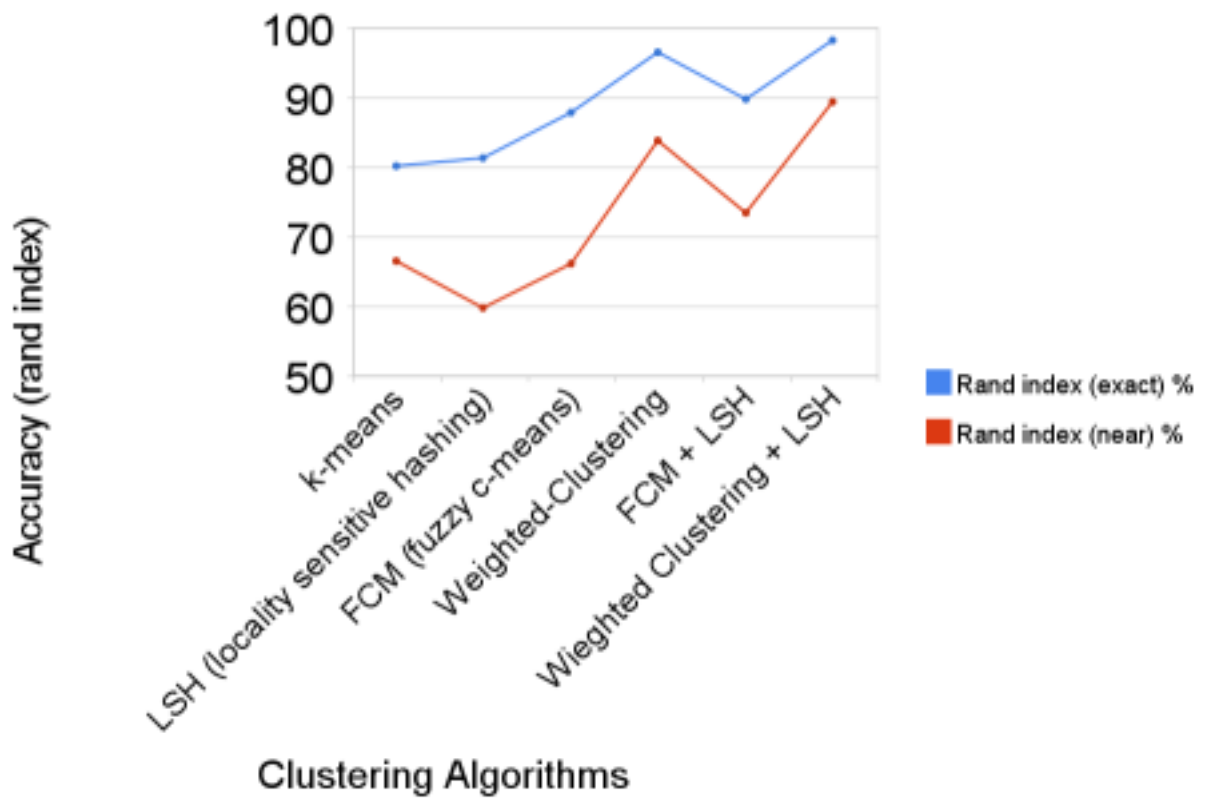


Figure 5.1: Final Results

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this work, we presented a novel technique to cluster real-estate listing from WWW. We even believe this technique can be used to cluster data with low-weights missing parameters. We have discussed the challenge behind good clustering technique for real-estate listing. Clustering data with missing values with imputation is as unreliable as wrong value. Therefore, we used *marginalization* which ignores missing values and compute partial distance on missing values. We have shown through a number of test that the proposed technique can be fully applied to retrieval and clustering of real-estate listing on commercial or large scale, as it is capable of clustering faster and efficiently. This also included various aspect of IR system. The tests have shown that both weighted and its LSH variant is capable of detecting duplicates even if there is missing data or minor change in value. Furthermore, they are also capable of detecting near-replica. Even if 25% of data is missing this

technique perform better than other discussed in this work. Nevertheless, the technique shows better results than the other techniques discussed in this work.

In some cases, it is important to acknowledge and decided weights carefully. Clustering results may vary if weight constraints goes different. However, we let system compute effective weights and let user decide basic range and basic biasing to the features.

Finally, the testing environment and dataset we choose is to ease human understanding. We believe that algorithm will produce very similar numbers as we found out in our experiments on a restricted dataset. We believe that the proposed technique can allow real-estate collection system to generate trends and forecast prices. We see it has great potential of extracting "expected" value of a real-estate listing and if more, then consumer may bargain over prices.

The important feature of this technique is that it do not try to figure out missing values, this enables it to perform just like as there complete datum. So, the goal of clustering real-estate listing with missing or differ values has been achieved in this work. In addition to this technique a detailed framework for IR system was also presented in this work.

6.2 Future Work

In future work, we recommend to compare proposed technique with Expectation-Maximization (EM). Although EM is a data imputation, we believe it worth comparing it with EM in certain aspects. There is also a need of observing

behaviour of proposed technique on vast variety of weights and thresholds. During our tests we found it very interesting that changes in weights can be led to different clusters. So, we believe there is a space to examine this technique over different weights.

During this work, we have been focused on unsupervised learning. Though we believe that there is a good space for supervised learning techniques. *Marginalization* can be easily applied to Neural Network or Hidden Markov Model.

Another good extension would be to use this technique and build a price and sale model for real-estate industry.

Bibliography

- [AV06] D. Arthur and S. Vassilvitskii. How slow is the k-means method? In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 144–153. ACM New York, NY, USA, 2006.
- [Ber97] H. Berghel. Cyberspace 2000: Dealing with information overload. *Communications of the ACM*, 40(2):19–24, 1997.
- [BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.
- [CGM03] J. Cho and H. Garcia-Molina. Effective page refresh policies for web crawlers. *ACM Transactions on Database Systems (TODS)*, 28(4):390–426, 2003.
- [DDGR07] A.S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM New York, NY, USA, 2007.

- [DFG99] Q. Du, V. Faber, and M. Gunzburger. Centroidal Voronoi tessellations: applications and algorithms. *SIAM review*, 41(4):637–676, 1999.
- [DH73] R.O. Duda and P.E. Hart. *Pattern classification and scene analysis*. New York, 1973.
- [EMT01] J. Edwards, K. McCurley, and J. Tomlin. An adaptive model for optimizing performance of an incremental web crawler. In *Proceedings of the 10th international conference on World Wide Web*, pages 106–113. ACM New York, NY, USA, 2001.
- [FPSSU96] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in knowledge discovery and data mining*. MIT press Cambridge, Mass, 1996.
- [GG92] A. Gersho and R.M. Gray. *Vector quantization and signal compression*. Springer, 1992.
- [GIM99] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *In Proc. 25th Internat. Conf. on Very Large Data Bases*, 1999.
- [HB01] RJ Hathaway and JC Bezdek. Fuzzy c-means clustering of incomplete data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 31(5):735–744, 2001.
- [HKY99] L.J. Heyer, S. Kruglyak, and S. Yooseph. Exploring expression data: identification and analysis of coexpressed genes, 1999.

- [HW79] JA Hartigan and MA Wong. A k-means clustering algorithm. *JR Stat. Soc., Ser. C*, 28:100–108, 1979.
- [JD88] AK Jain and RC Dubes. Algorithms for clustering data. *Printice-Hall, New Jersey*, 1988.
- [JMF99] AK Jain, MN Murty, and PJ Flynn. Data clustering: a review. *ACM computing surveys*, 31(3), 1999.
- [KH00] G. Karypis and E.H.S. Han. Fast supervised dimensionality reduction algorithm with applications to document categorization & retrieval. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 12–19. ACM New York, NY, USA, 2000.
- [MNU00] A. McCallum, K. Nigam, and L.H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178. ACM New York, NY, USA, 2000.
- [MRS08] C.D. Manning, P. Raghavan, and H. Schtze. *Introduction to information retrieval*. Cambridge University Press New York, NY, USA, 2008.
- [Sal71] G. Salton. The SMART retrieval system experiments in automatic document processing. 1971.
- [TCS⁺01] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie,

- R. Tibshirani, D. Botstein, and R.B. Altman. Missing value estimation methods for DNA microarrays, 2001.
- [TNA95] V. Tresp, R. Neuneier, and S. Ahmad. Efficient methods for dealing with missing data in supervised learning. *Advances in neural information processing systems*, pages 689–696, 1995.
- [VGCJ99] A. Vizinho, P. Green, M. Cooke, and L. Josifovski. Missing data theory, spectral subtraction and signal-to-noise estimation for robust ASR: An integrated study. In *Sixth European Conference on Speech Communication and Technology*. ISCA, 1999.
- [YR01] KY Yeung and WL Ruzzo. Principal component analysis for clustering gene expression data, 2001.

Warszawa, dnia

Oświadczenie

Oświadczam, że pracę magisterską pod tytułem; ”**Retrieval and clustering of real estate advertisements.**” , której promotorem jest **dr. Maciej Grzenda** wykonałem samodzielnie, co poświadczam własnoręcznym podpisem.

.....