HABILITATION THESIS

Knowledge extraction and processing approaches for structured and unstructured data

Assoc. Prof. Mihaela Dînșoreanu

Faculty of Automation and Computer Science
Technical University of Cluj-Napoca

2015

Table of Contents

Table List

Figures List

Knowledge extraction and processing approaches for structured and unstructured data.

Abstract

The present habilitation thesis presents the professional activity of the candidate and the scientific activity that was conducted after the defence of her PhD thesis at the Technical University of Cluj-Napoca on 9.07.2004 followed by the title confirmation by the Ministry of Education and Research Order nr. 445/2.08.2004. The research activity related to the PhD thesis was concerned with the design of agent models and multi-agent systems involving cooperating agents. The models were applied in an e-learning system. After finishing the PhD thesis, the candidate remained interested and active in the artificial intelligence field being involved in several national projects. Areas that were addressed in the early projects are: data mining solutions that were applied on medical data, knowledge extraction and representation solutions that were applied on archival documents written in natural language. More recently, the research interests of the candidate was focused on various aspects related to semantic data representation and integration, knowledge extraction by employing various machine learning techniques and knowledge processing.

a. *Data representation and semantic integration*.

The problem that was addressed by this work is the semantic alignment of data imported from different structured data sources (i.e. relational databases, XML files, CSV files), the automatic design of a unified integration data structure and the automatic design of the corresponding ETL processes. In this respect we developed a solution based on the data sources metadata, semantic synonymy relations from WordNet and the Jaro-Winkler lexical similarity algorithm. We developed a semantically annotated repository that is used in the semantic merging algorithm to produce information as accurate as possible for heterogeneous data integration and dynamic generation of ETL processes. We performed experiments on different sizes of data source structures in order to evaluate the scalability and performance of the solution.

b. *Knowledge extraction.*

In the problem of knowledge extraction our latest interests were focused on unstructured data sources, namely natural language texts. In this respect we aimed for topic identification in order to define context for context-sensitive recommendations. Another objective we addressed was opinion mining. For these objectives we explored several dimensions: supervised vs. unsupervised solutions, domain dependent vs. domain independent solutions, language specific solutions for English and Romanian. We proposed a unified topic model based on the Latent Dirichlet Allocation approach for defining the context and also a set of contextual and behavioural distance metrics in order to match content (recommendations) to context. For the opinion mining problem we proposed a processing flow and a learning approach based on an original set of meta-features of text elements (i.e. POS combinations as bigrams). The proposed supervised solution performed very well on English documents and we adapted the solution by employing the Romanian grammar rules to Romanian documents. Another approach that we proposed for the same opinion mining problem was an unsupervised one based on the double-propagation algorithm. We proposed an enhanced set of rules for the double-propagation and also a minimal set of very general seed-words (i.e. two) that proved to perform comparable with the state of the art results. We also addressed the problem of opinion mining from social networks, namely tweets. The challenges related to tweets are manifolds: tweets do not use a correct language in many cases, they have a

limited size (140 characters) and usually contain slang, abbreviations, special symbols, emoticons etc.
Normally an opinion includes the target (i.e. the entity that is referred), the opinion holder (who expresses the opinion) and the opinion itself. In the case of social network content or postings the holder is by default the current user but, although the posting involves a sentiment polarity, there is not always a target. The sentiment might be just a feeling. Supervised approaches are not that useful to classify tweets given the diversity of content existing in tweets. We investigated unsupervised approaches for polarity classification of tweets. We also analysed existing lexical, semantic and benchmark data resources that can help to increase the classification performance. We proposed a processing flow and applied our solution to four benchmark datasets in order to compare our results with other state-of-the art solutions.
c. *Knowledge processing*
We investigated methods to derive more information out of the extracted knowledge. In this respect we proposed methods for contradiction detection in opinions, for opinion driven community detection and for financial data streams mining. The first two methods are relying on the results of the opinion mining methods. The contradiction detection method identifies opinions on the same target and evaluates the polarity distance between them. It also identifies basic forms of negation in order to identify contradictory opinions. The community detection solution aims the identification of opinion holders that share similar opinions. The solution also integrates social network data and employs two methods for integrating multidimensional data: Network Integration and Partition Integration. We also experimented 9 multiple-target aggregated similarity functions that lead to the following conclusions: linear functions perform poorly for data sets with multiple targets and functions that calculate the average similarity have greater resilience to noise.
The proposed approaches and obtained results are detailed in the Scientific achievements chapter.

The future research interests of the candidate are shifting towards BigData methods applied in Internet of Things solutions. The envisioned challenges include:

- Semantic alignment of meta-data that describes data received from heterogeneous, dynamic data sources such as devices, sensors, etc.
- Develop efficient data representation models and processing methods, with a special focus on data streams
- Integrating machine learning techniques and semantic resources in data (streams) analysis in order to perform context-aware knowledge extraction

More details can be found in the Scientific future development plans chapter.

Professional and academic achievements

Teaching Activity

The teaching activity of the candidate started in 1995, covering the following **courses**: Object-Oriented Methods (2000-2009) for the Romanian and English BSc programs, Software Design (2010-current) for the Romanian and English BSc programs, Project management (2007-current) for the Romanian and English BSc programs, Software Engineering (2007-current) for the MSc program.

The teaching activity includes **laboratory works** at the following disciplines:
Functional programming (1995-1997), Programming techniques (1998-2002), Object-Oriented Methods (2000-2009), Software Engineering (MSc program).

The candidate has supervised an average of 7 diploma thesis/academic year and 2 masters thesis/academic year, since 2005.

Research Projects

Team member in SWARA- Mobile System for Rehabilitative Vocal Assistance of Surgical Aphonia (2014-2016) Project Type: PN-II-PT-PCCA-2013-4-1660. Contract no. 6/2014

Team member in Green Active Management of Energy in IT Service centres (GAMES) (2010-2012) Project Type: FP7-ICT-2009-6.3: ICT for Energy efficiency

Team member in Ontology Driven Automatic Web Service Composition (MAESTRO) (2007-2010) Project Type: CNCSIS PN II Ideas Nr. 333/2007

Scientific responsible in Integrated System for developing semantically-enhanced archive content (ArhiNET) (2007-2010) Project Type: CNCSIS PNII Partnerships Nr. 11051/2007

Scientific responsible in A High Performance System for Generation and Testing of Random Number Sequences for Cryptographic Applications (CryptoRand) (2007-2010), Project Type: CNCSIS PNII Partnerships Nr. 11-020/2007

Team member in Screening, Prophylaxis, And Correction of Congenital Malformations Genito-Urinary System in the Era of Minimal Invasive Therapeutic Procedures (Laparoscopy, Endoscopy) (SCANURGENT) (2006-2008), Project Type: CEEX VIASAN Nr.154/2006

Team member in Integrated IT system for assuring traceability and quality control in food industry (FOODTRACE) (2006-2008), Project Type: CEEX AGRAL Nr. 33/2006

Scientific responsible in Intelligent System for assisting the therapeutical decision at patients with prostate cancer (INTELPRO), (2005 - 2008), Project Type: CEEX-INFOSOC Nr. 18/2005

Scientific impact

h=7, according to Google Scholar Citations

h=5, according to Scopus
h=2, according to ISI Web of Knowledge

Reviewer

Journals

- International Journal On Advances in Intelligent Systems ISSN: 1942-2679
- International Journal of Web Information Systems ISSN: 1744-0084 (Emerald)
- Computers and Electrical Engineering, ISSN: 0045-7906 (Elsevier)
- Engineering Applications of Artificial Intelligence, The International Journal of Intelligent Real-Time Automation, ISSN: 0952-1976 (Elsevier)
- Romanian Journal of Information Science and Technology (ROMJIST) ISSN: 1453-8245

Conferences (Program Committee)

- International Conference on Information Visualization Theory and Applications (IVAPP) 2015, 2014
- International Conference on Adaptive and Self-Adaptive Systems and Applications (Adaptive) 2015 - 2009
- International Conference on Mobile, Hybrid, and On-line Learning (eLmL), 2014-2009
- IEEE International Conference on Systems, Man, and Cybernetics, 2014
- IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR) 2014, 2012, 2010, 2008
- Pre-ICIS SIGDSS Workshop on "Reshaping Society through Analytics, Collaboration, and Decision Support: Role of BI and Social Media", 2013

Professional membership

Member of ACM and IEEE

Member of the Special Interest Group on Decision Support, Knowledge, and Data Management Systems (SIGDSS)

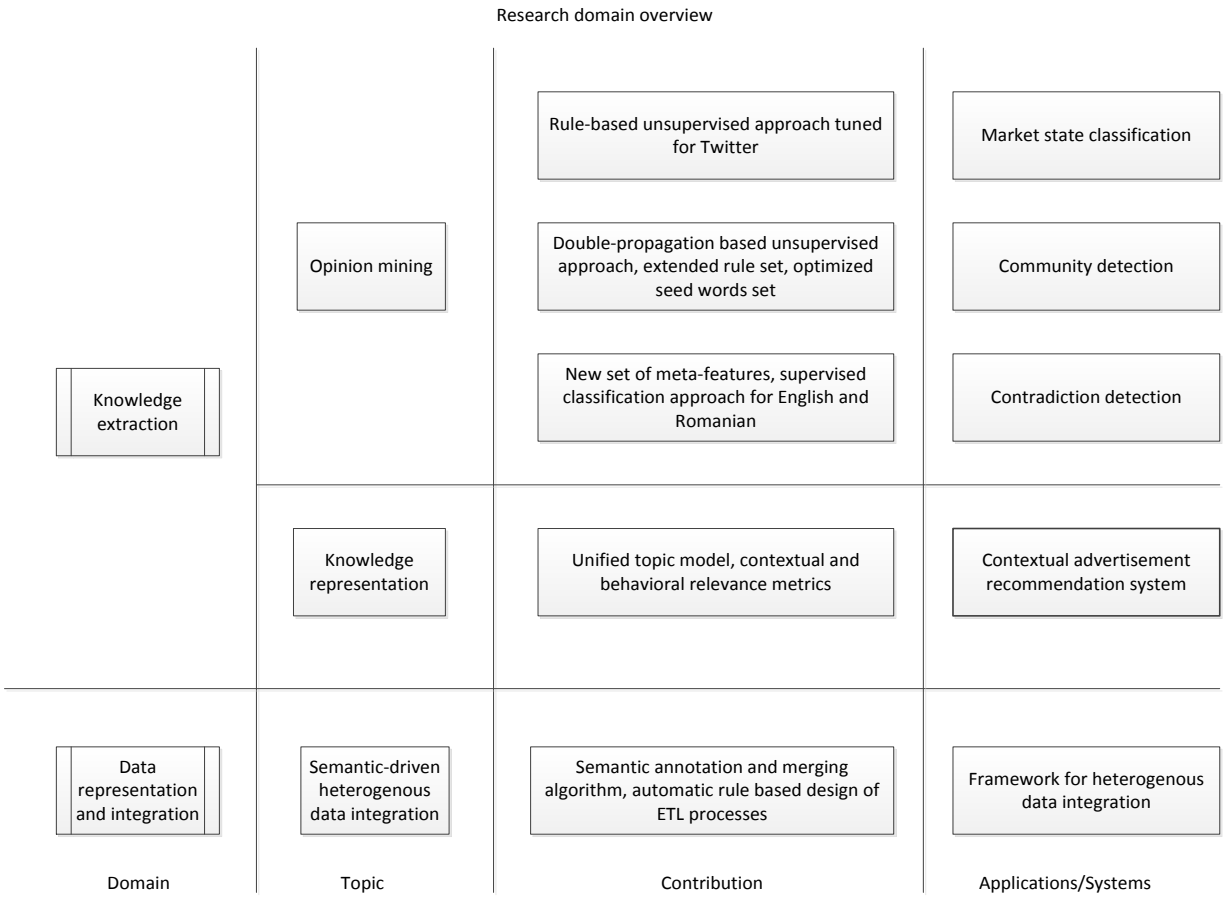
List of papers presented in the thesis

1. **Dinsoreanu, M**., Braescu, L., Bacu, A. (2012). Towards a semantic-driven automatic staging area design for heterogeneous data integration. In Proceedings of the 14th International Conference on Information Integration and Web-based Applications & Services (pp. 290-293). ACM.
2. **Dinsoreanu, M**., Potolea, R. (2013). Towards a Unified Thematic Model for Recommending Context-Sensitive Content. In Knowledge Discovery, Knowledge Engineering and Knowledge Management (pp. 68-83). Springer Berlin Heidelberg.
3. Hasna O.L., Macicasan F.C., **Dinsoreanu M.** and Potolea R. (2014), Sentiment Polarity Extension for Context-Sensitive Recommender Systems, In Proceedings of International Conference on Knowledge Discovery and Information Retrieval (KDIR 2014), (pp. 126 - 137), ScitePress

4. Anisiei, A. L., **Dinsoreanu, M**., Lemnaru, C., Potolea, R. (2014). Extracting opinion holders and targets in Romanian texts. In Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on (pp. 37-42). IEEE.
5. Russu, R. M., **Dinsoreanu, M**., Vlad, O. L., Potolea, R. (2014). An opinion mining approach for Romanian language. In Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on (pp. 43-46). IEEE.
6. Cosma, A. C., Itu, V. V., Suciu, D. A., **Dinsoreanu, M**., Potolea, R. (2014). Overcoming the domain barrier in opinion extraction. In Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on (pp. 289-296). IEEE.
7. Suciu D.A., Itu V.V., Cosma A.C., **Dinsoreanu M**. and Potolea R. (2014), Learning Good Opinions from Just Two Words Is Not Bad, In Proceedings of International Conference on Knowledge Discovery and Information Retrieval (KDIR 2014), (pp. 233 – 241), ScitePress.
8. **Dinsoreanu M**., Bacu A., (2014), Unsupervised Twitter Sentiment Classification, In Proceedings of International Conference on Knowledge Discovery and Information Retrieval (KDIR 2014), (pp. 220 – 227), ScitePress
9. **Dinsoreanu M**, Potolea R. (2014), Opinion-driven communities' detection, (M. Indrawan-Santiago, Eds.) International Journal of Web Information Systems, 10(4), 324–342.
10. **Dînşoreanu M**., Potolea R. (2013), A scalable approach for Contradiction Detection driven by Opinion mining, In Proceedings of International Conference on Information Integration and Web-based Applications & Services - IIWAS '13 (p. 7–15).
11. Ionescu V., **Dinsoreanu M.** (2013), An approach to mining financial markets through market state classification, In 2013 IEEE 9th International Conference on Intelligent Computer Communication and Processing (ICCP) (p. 43–46) IEEE.

Scientific achievements

The scientific areas of interest are structured in the following schema:



1. Data representation and integration

1.1 Semantic-driven heterogeneous data integration

1.1.1. INTRODUCTION

We experience nowadays an exponential increase in data flow and data volume in the context of information gathering. Corporations store their business information in distributed, heterogeneous data storages such as relational databases, spreadsheets, XML documents and various other flat files or data blobs. Each of these aforementioned data sources have their own different structure and format making an integrated data analysis difficult. That is why the problem of integrating heterogeneous data from distributed data sources in a single data storage unit is relevant. Our approach is presented in (Dinsoreanu, 2012).

The concept of heterogeneity we are addressing refers to data sources that employ lexical differences in the data representation (such as the usage of acronyms, abbreviations and synonyms) while the semantic of the data is the same. Moreover, we consider also structural heterogeneity that is related to representing the same concept using different data types or data structures. In order to assert and strengthen the semantic relationship between related data from different storage types, we aimed to integrate large semantic and lexical repositories available online, with lexical and structural similarity models applied on the source metadata.

Moreover, we propose an approach for the problem of the semantic alignment of data stored in independent, heterogeneous data sources while automatically designing the integrated destination schema and the corresponding Extract-Transform-Load (ETL) processes. Our approach aims at the development of a semantic repository starting from multiple heterogeneous data sources, in type and schema, using available metadata, semantic and lexical information, through a semantic merging algorithm. The obtained semantic repository allows the automatic generation of the integrated destination (Staging Area) schema and of the ETL transformations.

Based on our approach, we developed a framework for heterogeneous data integration that relies on a set of customizable and extensible rules and operations used for designing automatic ETL processes, in conjunction with the semantic information conveyed by the semantic merging algorithm used.

The data integration problem has been of high interest especially since huge amounts of data have become available via the Web. Some of the related work is focused only on the semantic alignment of the heterogeneous data sources while other approaches address also the design of the target, integrated structure and the corresponding ETL processes. In (Beneventano & Bergamaschi, 2004) the authors present a framework for semi-automatic information extraction and integration of heterogeneous information sources called MOMIS. The framework generates an integrated schema that unifies the underlying structures in terms of classes. It also builds the domain ontology as the synthesis of the integration process. However, the framework does not address the transformation processes to migrate the source content into the unified structure.

The approach presented in (Zieg, 2007) is specific in terms of handling data integration based on the user-specific perspective. This approach is applicable in heterogeneous data receivers contexts such as flexible cooperation between businesses; virtual, dynamic organizations etc. A semantic multidatasource language to declaratively manipulate so-called IConcepts is proposed. IConcepts are conceptual building blocks that link to the same real-world concept by data providers. Therefore, the user input is crucial and it includes declarative statements formulated in the semantic multidatasource language. Our approach requires less user intervention.

An extended method to map ER structures to RDF is presented in (Chatterjee & Krishna, 2007). The method addresses the mapping of elements such as aggregation, specialization, multi-valued attributes. The resulted RDF triples are represented as directed graphs so the data integration problem becomes a graph merging problem. The approach is limited in the sense of the source data structure that is RDB.
In (Hao, 2005) the authors investigate how AutoMed metadata can be used to express the schemas present in a data warehouse environment and to represent data warehouse processes such as data transformation, data cleansing, data integration, and data summarization. The approach also shows how evolution of both data source and data warehouse schemas can be handled.
More recent work (Kavitha, Sadasivam, & Shenoy, 2011) presents another approach for integrating heterogeneous databases by creating a local ontology for every database mapping the physical quantities of database to the physical quantities of ontology. Finally, ontology matching between local ontologies should be done to resolve the problem of schema matching among the heterogeneous databases. The proposed approach is also limited to the relational data sources and doesn't consider the design of the ETL processes.
(Skoutas, Simitsis, & Sellis, 2009) illustrates a customizable and extensible ontology-driven approach for the conceptual design of ETL processes. The approach relies on a graph-based representation as a conceptual model for the source and target data stores. The flows of ETL operations are devised by means of graph transformations which are determined by the semantics of the data with respect to attached domain ontology. In this approach the source and target structure as well as specific domain ontology is considered as available and accurate.
The particularities of our approach can be summarized by the following features:

- metadata from various data sources (relational databases, CSV files, Excel spreadsheets, XML files) is aligned against a general purpose taxonomy (WordNet), thus no specific domain ontology is needed
- all framework components (i.e. data source schema and data, destination structure and in-between ETL processes) are aligned semantically, which allows for consistent evolution.

We evaluate the scalability and matching performance of our approach using a set of heterogeneous data sources scenarios, created based on sample database tables and files. Our preliminary findings show that the proposed approach can be used with success even for large data sources and ETL scenarios.

1.1.2. THE CONCEPT OF OUR SOLUTION

Our solution for dealing with heterogeneous data integration aims at providing an automated process by which users can semantically align data from various operational data sources into one semantically consistent structure.
Data matching is an important component during Data Integration and addresses the degree of similarity between two data entities from different data sources. Matching data entities can be done at various levels and can be based on various decisional processes. We considered some of the possible relationships between entities ranging from simple relationships such as synonymy and lexical matching to more complex ones like structural matching and semantic matching in terms of hyponymy, hypernymy, aggregation or composition.
The processing done by our solution is centred on three main data repositories:

- Metadata Repository – stores and manages the metadata nodes for each data entity related to a specific operational data source, thus each data source has its own Metadata Repository available in memory.

- History Metadata Repository - wraps up all the Metadata Repositories for all operational sources processed so far; allowing matching data entities from distinct data sources.
- Semantic Repository - stores and manages the semantic concepts obtained from general purpose ontologies and mapped onto metadata nodes from the Metadata Repository thus creating an Enhanced Metadata Repository.

The overall architecture of our system is presented in Figure 1.

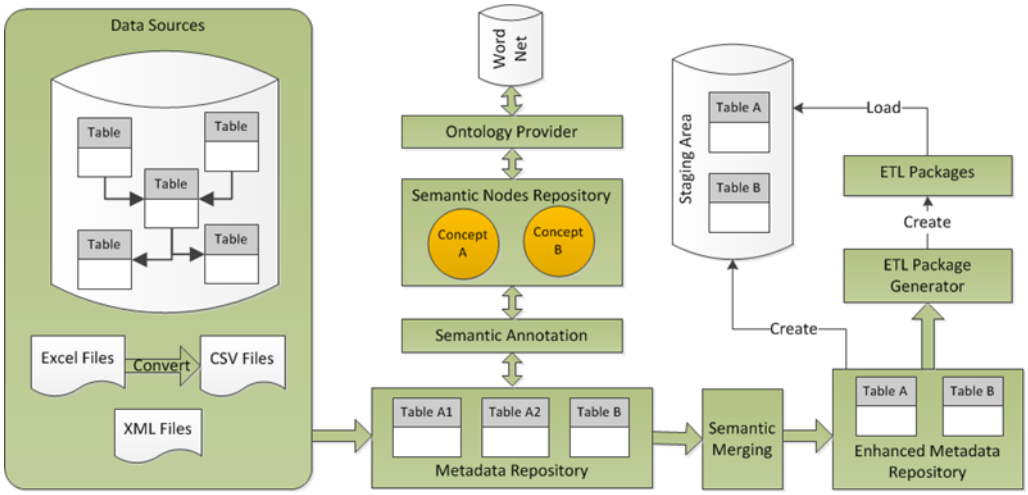


Figure 1 Overall architecture

1.1.2.1 Operational data sources

In our analysis we covered the top four most used data source types and included them as main inputs for our system's processing flow: Relational and non-relational databases, CSV (comma-separated values) files, Excel spreadsheets, XML files.

The concepts and algorithms presented in the next sections will be exemplified on relational data sources because it provides the most valuable metadata information required for our approach. In order to obtain the knowledge to be included in the source metadata repository, the input files are pre-processed differently, depending on their type:

- Tables containing database table relationships (i.e. between referencing and referenced tables) and ETL process generator configuration data (i.e. configuration filters, data connection strings and ETL package variables) are created and populated
- The Primary and Foreign key columns and number of records are stored for all repository tables
- For each Primary and Foreign key column, the referencing and referenced columns from other tables are stored

Considering the metadata repository knowledge, tables will be tagged as Stand-alone (i.e. table doesn't reference any table, is only referenced by other tables), Intermediate (i.e. table references only Stand-alone tables) and Linkage (i.e. table references and is referenced by any other type of table).

For CSV and XML data source types the pre-processing phase consists of extracting metadata information and storing it in the source metadata repository.

1.1.2.2 The Ontology Provider and Semantic Model

In order to obtain a unified metadata model, the semantic data needs to be accessed and used in our annotation processing flow from the perspective of each of semantic relationship type. A straightforward solution was to wrap the ontology component that handles semantic data into an ontology provider that streamlines the needed semantic information based on a well-

defined contract complying to needed or required semantic relationships, allowing for transparent usage of a multitude of semantic information providers.
The semantic model we define contains several elements. The entity that represents the core of the semantic annotation algorithm is the Semantic Repository (SR). It is a set of Semantic Nodes (SN) representing structures that contain Concepts, mergeable tables (mTables) and mergeable columns (mColumns).

$$SR=\{SN_1, SN_2,\ldots,SN_n\} \quad (1.1.1)$$
$$SN= [Concept, mTables, mColumns] \quad (1.1.2)$$

Each Concept has a name and a list of synonyms. Two or more different data source tables that are mapped to the same Concept through synonymy will be merged into a single table in order to keep data and schema consistency.
We rely on WordNet as the main source of semantic data but our architecture supports an Ontology Provider that can be connected with multiple ontology sources.
WordNet, as a large lexical database of English, contains nouns, verbs, adjectives and adverbs that are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. From the relations available, the synonymy is considered as the most important and will be used in our approach. Actually, a step of the semantic annotation algorithm is Concept synonym extension. This process will check each Concept by WordNet and obtain all its synonyms. For example, for a Concept named "Client" from the Semantic Repository, "Customer" and "Guest" are added to its collection of synonyms.
In order to take into consideration the heterogeneity that arises in databases when naming conflicts occur due to the use of abbreviations, a lexical similarity algorithm is used to determine if two or more tables represent the same Concept. The lexical similarity algorithm is based on the Jaro-Winkler distance (Winkler, 1990) that is designed and best suited for short strings. The score is normalized such that 0 equates to no similarity and 1 is an exact match. For obtaining improved lexical and semantic matching results we consider the following pre-processing tasks, based on empirical analysis: the uppercase is used when comparing table and column names; table prefixes like "TB_", "TST_" etc. are trimmed in order to increase the similarity for lexical matching and allow better synonym extension by WordNet. To avoid table name matches through WordNet to ripple at column level the table name is trimmed from its column names where needed. For example, for two tables named "TB_Dept" and "Department", the similarity score given by the Jaro-Winkler approach is 0.86, therefore the semantic annotation algorithm will consider these tables mergeable and map them to the same Concept named "Department".
A similarity threshold of 0.85 is chosen after running a considerable amount of test scenarios.

1.1.3. THE ALGORITHMS

The most relevant algorithms are presented next.

1.1.3.1 The Semantic Annotation Algorithm

The semantic annotation process is done via the Ontology Provider component by using WordNet in order to ensure semantic alignment of processed data entities and easier access and availability of ontology services. Extensibility is a key feature in our approach thus the data entity matching process is designed to support additional matching definitions and the Ontology Provider is designed taking into consideration multiple ontology sources and supports this feature even if for simplicity we use only one ontology source as proof of concept. The semantic annotation algorithm is presented below:

Semantic Annotation Algorithm

```
void Annotate(List of ISemanticEntity metadata)
{
      TrimMetadata(metadata);
      oProvider := new OntologyProvider();
      repository := SemanticRepository.ConceptsRepository;
      for each entity in metadata
      {
            //check if any existing semantic nodes match //the entity
            for each semanticNode in repository
            {
                  isMatch := Match(entity, semanticNode);
                  if (isMatch)
                  {
                        //if a match is found map the entity to
                        //the concept
                        entity.MappedConcept := semanticNode;
                        semanticNode.AddSemanticEntity(entity);
                  }
            }
            if (entity.MappedConcept == NULL)
            {
                  //if no concept is mapped to the entity
                  newSNode := CreateSemanticNode(entity.EntityName);
                  repository.Add(newSNode);
                  //map the new node to the entity
                  entity.MappedConcept = newSNode;
                  newSNode.AddSemanticEntity(entity);
            }
      }
}
```

The core of the semantic annotation process is the Match function responsible to pair up related data entities from the underlying data sources via Metadata Nodes. A Metadata Node (ISemanticEntity) may contain table or column information. The Match algorithm pairs each data entity in a Metadata Node with a semantic concept from a Semantic Node obtained through the Ontology Provider. The Match function between a Metadata Node and a Semantic Node yields true in the following cases:

- the base concept and entity name are equal
- there exists a base concept synonym equal to the entity name
- the entity name is matched lexically by the Jaro-Winkler matching algorithm with at least one of the existing metadata entities name mapped on the same base concept
- the entity name is matched lexically by the Jaro-Winkler matching algorithm with the base concept's name

1.1.3.2 The Semantic merging module

The semantic merging algorithm considers the semantically annotated metadata knowledge obtained from the considered data sources and will obtain the integrated schema of the Staging Area destination structure together with all the mappings required for the ETL processes automation. Merging of semantically enhanced metadata nodes is done based on a weighted average matching statistic generated from a 3-point perspective concerning

semantic similarity (synonymy), lexical similarity and structural similarity. The weights of this compound matching process have been determined empirically at this point in the design. After the merging algorithm has been executed the metadata repository contains all the required information in order to produce inter-schema mappings from the sources to the destination data store used to generate ETL packages to populate the homogeneous Staging Area structure with data from the heterogeneous data sources and thus providing valid, enhanced data integration. The semantic merging algorithm is presented next:

Semantic Merging Algorithm

```
void Merge(List of Table metadata, List of MergeableTables mergeables)
{
      for each mergeableTables in mergeables
      {
            rootTables := mergeableTables;
            overlappingMergeables  :=  GetOverlappingMergeables(mergeables,
            rootTables);
            toMerge := GetTablesToMerge(overlappingMergeables, rootTables);
            merged := GetMergedTable(toMerge);
            for each table in toMerge
                  table.SetDestination(merged);
      }
      unmapped := GetUnmappedTables(metadata);
      foreach (table in unmapped)
            table.SetDestination(table);
}
```

This semantic merging algorithm's processing flow can best be described as the efficient utilization of the semantically enhanced metadata provided by the semantic enhancement module in order to create and establish destination structures for each of the data entities from the underlying operational sources. This is done by rippling the processing from metadata level to table-structure level to column-structure level.
The semantic merging algorithm's most important subcomponent is the Table-level merging component and we have detailed it below:

Table-level Merging Algorithm

```
Table GetMergedTable(List of Table toMerge)
{
      merged := AggregateTableMetadata(toMerge);
      merged.Columns := GetMergedColumns(toMerge);
      allColumns := AgregateColumns(toMerge);
      for each column in allColumns
      {
            destColumn := GetMergedDestination(column, merged);
            if(destinationColumn != NULL)
                  column.SetDestination(destColumn);
            else column.SetDestination(column);
      }
      return merged;
}
```

1.1.3.3 ETL Mappings and Transformations

The design of the Extract-Transform-Load (ETL) processes represent the final step used by our approach in order to actually integrate data stored in the considered heterogeneous sources. In the Business Intelligence field, one of the main challenges during the early steps of a data warehouse project represents the identification of the appropriate transformations and the specifications of inter-schema mappings from the source to the destination data stores.

In our approach we consider multiple sources and one destination with an integrated homogeneous structure. This represents the Staging Area and is automatically generated against the considered taxonomy. It is created based on the merged data source schemas provided in the semantic merging algorithm output. Practically, for each collection of mergeable tables, one Staging Area table is created as containing the entire list of specific and mergeable columns, a row number column and a column that specifies the record data source table. This is done in such a way to preserve data and schema consistency and also the interoperability among the data sources.

In the following, the focus is set on the means used for transforming the knowledge obtained by the combined semantic algorithms into a customizable and extensible set of rules and operations that govern the automatic design of ETL processes. More specific, for a created ETL process design, multiple instances i.e. ETL packages are configured individually based on the knowledge established in the metadata pre-processing phase. This is done transparently from the user's perspective.

The enhanced metadata repository represents the first important output of the approach described so far, used in designing and configuring ETL processes. For each table considered as a source, the repository contains its Staging Area table destination along with the list of columns and their respective destination mappings. All the table and column mappings compose an essential rule in the generation cycle of an ETL package.

The second input for ETL processing is constructed based on the set of mapping rules described above. It represents the set of operations or transformations that occur between the source and target data stores such as source extraction script definition, data auditing (row counts), data type conversion and destination loading specifications (data access mode, commit size etc.).

The automatic generation cycle for an ETL package is composed of the following steps:

1. A package template is chosen for the appropriate data source type
2. Template configuration data are mapped onto the matching data records from the Configuration table
3. Inter-schema mappings from the data sources to the Staging Area are inserted in the package data flow
4. Column data type conversions and row count auditing logic are updated
5. Error and execution logging information are configured

For example, tables "Client" and "Customer" are considered, each having a different data source, independent of type. The input for ETL design represents the semantically annotated metadata of the merged table "Customer". Both input tables are labelled as "Linkage", thus the destination will have the same label, causing no referential and type conflicts. The ETL generator logic will decide that two ETL packages are necessary, having source extraction scripts mapped on the metadata of the two tables, the required data type conversions, fast load data access mode and maximum commit size for the same destination – the merged table "Customer".

For the file types considered as data sources by our approach, the destination will be a table constructed from the gathered metadata transparently handled, as explained above. The ETL

packages corresponding to the same data source can be parallelized at runtime, reducing total time to the slowest package running time, which is a large improvement for the time consuming data integration process.
Regarding the generated Staging Area structure presented below, table relationships are missing in order to increase the performance of queries for further data warehouse ETL extractions.

1.1.4. CASE STUDY

The following case study illustrates the criteria for choosing data sources as well as how the degree of heterogeneity is measured. We performed multiple experiments that include three categories of scenarios containing a varying number of database tables: small scenario (less than 15 tables), medium scenario (between 15 and 40 tables), and large scenario (more than 40 tables). For demonstration purposes, we will present the small scenario that is comprised of two database sources each with 12 tables and an average of 3 columns per table. The structures of the two sources A and B can be seen in Figure 2.
With expert validations of table mappings we managed to obtain a Staging Area with 15 tables and an average of 5 columns per table.

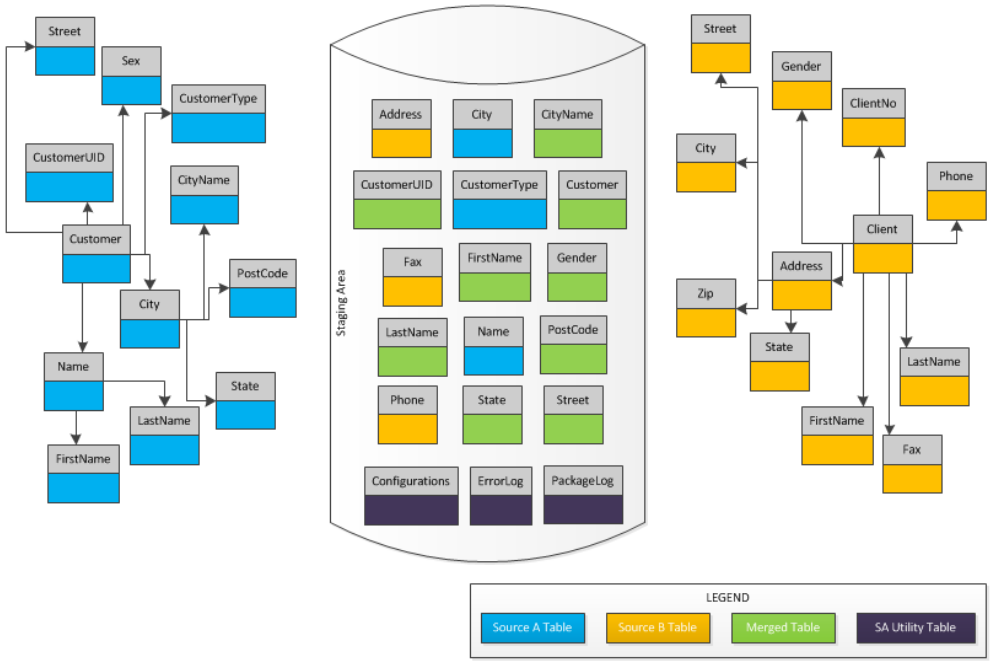


Figure 2 Case Study

The Staging Area columns are slightly increased due to extra columns being created by the generator in order to maintain database referential integrity such as 'RowNumber' and 'SourceEntity' columns for each table. By excluding these columns we achieve an average of 3 columns per table for the Staging Area. The column ‘RowNumber’ is used for handling primary key constraint violations consisting in clashes between the same primary key column values that are to be inserted from multiple tables that are merged by the proposed algorithm. The ‘SourceEntity’ column is used for specifying the data source entity from where the row data was inserted. It can be proven very useful for data warehouse ETL processing.

1.1.5. EVALUATION

From the data presented in the tables above several observations can be made regarding semantic processing.

The trimming of prefixes from table names allows synonym matching to occur between otherwise distinct strings such as 'AW_Client' and 'TB_Customer'.

The Jaro-Winkler lexical matching algorithm performs well when the compared string have a similar stem/root thus finding that 'Product' and 'TB_ProdCateg' along with 'SRC_ProductModel' and 'ProdCateg' are matched.

Taking into consideration the fact that the merging at table level has the greatest impact on the structure of the generated inter-schema mappings our approach outputs these mappings to the user for validation, thus potentially reducing errors introduced by the lexical matcher. The trimming of table name from column names allows synonym matching to occur between otherwise distinct strings such as 'ClientGender' and 'CustomerSex'. The trimming of prefixes from column names allows synonym matching to occur between otherwise distinct strings such as 'Contact' and 'HR_Liaison'.

The Jaro-Winkler lexical matching algorithm performs poor when the compared strings do not have a similar stem/root thus finding that 'FirstName' and 'CustomerFName' are not matched.

Validation for these cases is impractical thus we accept that in cases such as these the destination table will contain both columns when created. After user validation the tables will get merged based on the provided mappings.

An important task while striving towards obtaining a unified metadata model is dealing with the need of evaluating the performance of the unification process as each improvement made in data integration had a great impact on the unification process's outcome. To illustrate this fact further we will present the means by which we have evaluated and compared our various merging strategy outcomes throughout the development process of our framework.

1.1.5.1 Performance evaluation

In order to evaluate merging of Table and Column entities we present various merging decisions taken by our algorithm based on a variety of data inputs.

A Table entity is a data unit describing all the gathered metadata from our metadata extraction modules in regards to a table structure - be it a database table or a table structure defined in an XML.

A Column entity is a data unit describing all the gathered metadata from our metadata extraction modules in regards to a column structure - be it a database table column or a table column structure defined in an XML.

The weights and thresholds for each matching type are presented in Table 1 for reference and are used in the computation process of the overall merging decision in order to ensure overall matching accuracy.

Table 1 Merging Weights and Thresholds

Matching Type	Composite Weights	Threshold
Semantic	0.3	N/A
Lexical	0.2	N/A
Structural	0.5	N/A
Table Lexical	N/A	0.85
Column Lexical	N/A	0.91
Overall	N/A	0.65

First we present merging results at Table level in Table 2. Merging at this level has a greater impact on data integration thus matchings obtained in this step are forwarded to the user for cross-validation alongside the computed matching weight in order to provide aid to the user towards making an informed decision.
Second we present merging results at Column level in Table 3. Merging at this level does not have a significant impact on the data integration process and due to the possible multitude of column mappings that are generated and analysed by our framework's processing flow it would be impractical to forward them towards user validation. Thus column-level mappings are accepted as they are computed by the enhancer and merger.

Table 2 Table level merging decisions

Source A Table Name	Source B Table Name	Semantic Match	Lexical Match	Enhancer Decision	Structural Match	Merger Decision	Valid
TB_Customer	AW_Client	1	0.52	Invalid	1	0.9-Valid	True
SRC_DEPT	Department	0	0.86	Valid	1	0.67-Valid	True
TB_Cust	SA_Customer	0	0.9	Valid	1	0.68-Valid	True
Product	TB_ProdCateg	0	0.9	Valid	1	0.63-Valid	True
Contact_Type	Adress_Type	0	0.55	Invalid	0	0.11-Valid	True

Table 3 Column level merging decisions

"Client" Table Attributes	"Customer" Table Attributes	Semantic Match	Lexical Match	Enhancer Decision	Structural Match	Merger Decision	Valid
Contact	AW_Client	1	0.00	Valid	1	0.8-Valid	True
FirstName	Department	0	0.32	Invalid	1	0.56-Invalid	False
ClientType	SA_Customer	0	0.9	Valid	1	0.68-Invalid	True
Product	TB_ProdCateg	1	0.73	Valid	1	0.93-Valid	True
Contact_Type	Adress_Type	0	0.55	Invalid	0	0.11-Invalid	True

1.1.5.2 Scalability Analysis

In order to assess the performance of our proposed semantic algorithms we have devised a series of scalability tests as described in the following sections.
The scalability tests were ran on a machine with an i7CPU and 8 GB RAM memory to ensure optimal result gathering. That is why the execution times measured by our experiment fall in the range of 0-20 seconds. Even so the observed evolution of the execution times should be similar on most machines.
The scalability test scenario we considered processes a range of 10 to 1000 tables with an average of 3 columns per table and with an increment of 10 per step for a total of 100 iterations. The semantic entities (i.e. the columns and tables) were generated randomly and matches between entities were generated using a cloning interface. Execution time of the respective algorithms was measured for each step, being considered the most relevant performance factor.
Considering that most real-world scenarios involve sources with an average count of 10-100 tables and an average count of 5-10 columns per table then it is safe to assess that our semantic annotation and merging approach should be able to handle most of these processing cases.

Our approach's main processing advantage is the fact that annotations and semantic information is stored additively after each source processing. This means that for handling more than 2 sources no redundant processing will take place while running these algorithms to ensure current concepts are matched with previous ones. This provides great support for batch processing scenarios and from the algorithm's perspective there is no difference in performance between processing a source of 1000 metadata nodes (columns and tables) and two sources with 500 metadata nodes each. The difference between these two cases lies however in the pre-processing steps required to gather and obtain these metadata nodes, which is dependent on the source types used and the time required to access them.
The annotation algorithm performance test comprised of processing a range of 40-4000 semantic entities and mapping them onto 40-4000 semantic nodes. This simulated a worst case scenario for the enhancement algorithm due to the 0% merging load.
Regarding the annotation algorithm's performance we note that a slight increase of 2s in execution time corresponds to an increase of the workload by an average of 180 tables. The execution time of the semantic annotation algorithm is presented in Figure 3.

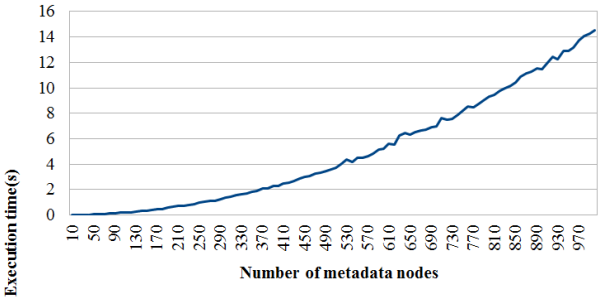


Figure 3 Semantic annotation scalability

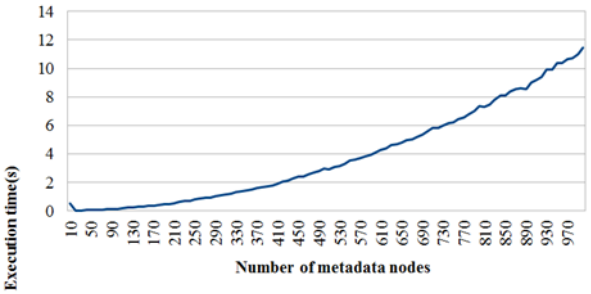


Figure 4 Semantic enhancement scalability

The merging algorithm performance test consisted of a 50% constant merging workload at table level and 100% constant merging workload at column level. That is, from the amount of table metadata nodes considered half of them were duplicates, and for each pair of tables to be merged all of their columns needed merging via a 1:1 relationship.
Similarly, the merging algorithm's performance can be summed up by observing that an increase of 2s in execution time corresponds to an increase of the workload by 160 tables on average. The execution time of the semantic enhancement algorithm is presented in Figure 4.

We can conclude that our proposed semantic annotation algorithm's worst case running time is O(n*k) where n is the number of metadata nodes in the Metadata Repository and k is the number of semantic concepts in the Semantic Repository. Runtime grows linearly as the number of metadata nodes in the Metadata Repository increases or the number of semantic

concepts in the Semantic Repository increases. The advantage in this annotation process is that in most cases, especially in large scenarios, k is significantly smaller than n and the running time strays away greatly under the worst case running time as it can be seen in Figure 3.
Our proposed semantic merging algorithm's worst case running time is O(n*c2)where n is the number of table metadata nodes in the Metadata Repository and c is the average number of column metadata nodes for each processed table metadata node.
So running time grows linearly as the number of metadata nodes in the Metadata Repository increases or the average number of columns per table in the underlying data sources increases. The advantage in this merging process is that in most cases, especially in large scenarios, c is significantly smaller than n and the running time strays away greatly under the worst case running time as it can be seen in Figure 4.
As a conclusion from the running time analysis there is still areas to improve regarding our proposed algorithm's performance but the processing output required from them is done in a reasonable amount of time considering the available technology available at the this time.

1.1.6. CONCLUSIONS AND FUTURE WORK

We have explored the use of data sources metadata, semantic synonymy relations from WordNet and the Jaro-Winkler lexical similarity algorithm, combined in a semantically annotated repository used in the semantic merging algorithm to produce information as accurate as possible for heterogeneous data integration and dynamic generation of ETL processes.
Also, we have proposed a formal means for an easy configuration and user transparent design of the ETL packages, which drive the construction of the ETL process, in conjunction with the semantic information conveyed by the semantic merging algorithm. Finally, we have evaluated our approach and we have proved its applicability in real-world settings.
Our future plans include the optimization of the approach and especially the extension of its semantic nature. Of a great interest is the direction to study the enhancements brought by a larger set of semantic relationships between entities and attributes, combined with the use of other large information providers.

2. Knowledge extraction from unstructured documents

One of the most challenging tasks we addressed is the information extraction from unstructured documents, namely natural language texts. We started by analysing solutions for context modelling and context-to-content matching to be applied in recommendation systems. In order to develop context models, our approach aimed to identify and classify topics in documents and we experimented with different similarity functions for identifying the best matching approach. Our next step involved opinion mining solutions. In this direction we started with supervised approaches applied in various types of documents written in English. Our goal was to shift towards unsupervised, domain-independent approaches and investigate cross-language possibilities.

2.1 Content-to-Context matching based on topic models

2.1.1. INTRODUCTION

Since we live in an information centric society, we are flooded with data (the so called “deluge of data” as The Economist called it in 2010) yet we still struggle to filter the information that we need.

In this context, an automatic identification of relevant pieces of information for each specific user is paramount. The goal is not only to expose to the user only meaningful information (relevant and of interest), but more important, at the right time and in the right context (Garcia-Molina, 2011).

Considering the above mentioned challenges we propose a generic model capable of providing context-sensitive content based on the underlying thematic similarity between an analysed context and the recommended content (Dinsoreanu M. P., 2013). Our approach is based on a unified topic model based on which the topics describing both the context and the content are extracted. This process is fuelled by the portions of the analysed entities having the highest descriptive value. Once these entities are annotated with thematic information, their reciprocal affinity, within the unified topic model, can be measured. Using such values, a topic based coverage aims to improve diversity and achieve serendipitous recommendations.

To summarize, the main steps of our process are:

- the extraction of the highest descriptive valued n-grams among the analysed entities;
- attaching thematic information to the analysed entities;
- maximizing diversity of the recommended content.

We apply our approach, as proof of concept, in the Online Advertising problem of the Best Match (OABM) (Broder & Josifovski., 2011) between a web page (active context), advertisements (suitable content) and the user that is currently interacting with that context (dynamic context). We claim that this problem can be mapped on our model by using a double instantiation of the context-to-content similarity relation. One instantiation describes the relation between a web page and an advertisement. The second has the same mapping for the content but describes the context as being the user, moreover his/her historical information. The combined triple recommendation between an active context, the content and the dynamic context is constructed by further processing the two instantiations.

Most of the reported approaches start by describing the matching content with relevant keywords. These keywords are compared with the descriptors of the ads (bid phrase) hence obtaining a lexical similarity (Manning, 2008), (Yih, 2006). Such an approach follows a pipeline with a few, well-defined stages. A pre-processing stage aims to prepare the content by cleaning, removing stop words, stemming and extracting some keyword candidates (words from the context, annotated with some descriptive features). Then the annotated keyword candidates are processed, in a Monolithic Combined approach, by a binary classifier. This is how the keywords are selected and the keyword selection step is completed.

Such an approach is generally enhanced with additional models that sustain the semantic similarity between web pages and advertisements (Broder A. F., 2007), (Zhang, 2008), (Ribeiro-Neto, 2005). This association generates a semantic score which, combined with the lexical score, consolidates the match. This semantic information can be embedded in a taxonomy (Broder A. F., 2007) and used to score the similarity based on the distance to the least common ancestor, if both the context and the advertisement can be mapped on it.

The third aspect to be considered in such a model consists of the particularities of an actual user. The associated historical information, if present, will influence the final match (Ahmed A. L., 2011), (Chakrabarti, 2008). User information can be attached to the advertisement or to the page (Chakrabarti, 2008) but, recent research explores the idea of user interest and behavioural trend (Ahmed A. L., 2011). Such a model can extract the dynamics of behaviour and make better recommendations.

The concept of a “topic” is described using a specialized mixed membership model called topic model. Such a model describes the hidden thematic structure (Blei D. N., 2003)in large collections of documents. The Latent Dirichlet Allocation (LDA) (Blei D. N., 2003) is such a

topic model. Its distinguishing characteristic is that all documents in the collection share the same set of topics, but each document exhibits those topics with different proportions (Blei D. , 2011). The topics are defined by a distribution over the whole set of available words (within the document corpus). The documents are described by a distribution over topics based on the used words. In a model like that, the only observable data are the documents' words. The topics, their distribution in documents and the distribution of words between topics need to be inferred. Direct inference is not tractable so approximation techniques are used (Heinrich G. , 2009).

Many systems produce highly accurate recommendations, with reasonable coverage, yet with limited benefit for practical purposes due to their "obviousness" hence lack of novelty. In this respect new dimensions that consider the "non-obviousness" should be considered. Such dimensions are coverage (percentage of items part of the problem domain for which predictions are made), novelty and serendipity, dimensions for which also (Ziegler, 2005) advocates. Since serendipity is a measure of the degree to which the recommendations are presenting items that are attractive and surprising at the same time, it is rather difficult to quantify. "A good serendipity metric would look at the way the recommendations are broadening the user's interests over time" (Herlocker, 2004), therefore the need for introducing timing and sequence of items analysis for RS arises again.

2.1.2. THE UNIFIED MODEL FOR STRUCTURED CONTENT REPRESENTATION

2.1.2.1 The Concepts and Terms used

In the following we formally define the notions used throughout the work.

- A *word* is the basic unit of discrete data (Blei D. N., 2003), defined to be an element of a vocabulary V;
- A *topic* $\beta_t = \{ p(w|t) \mid w \in V\}$ is a probability distribution over a finite vocabulary V of words w, where $\sum_{i \in \beta_t} i = 1$. Let T be the set of all topics;
- A *document* $d = \{w_1, w_2, \dots, w_N\}$ is a sequence of N words. Each document has an associated distribution over topics $\theta_d = \{p(t|d) | t \in T\}$ where $\sum_{i \in \theta_d} i = 1$;
- A *keyword* k_d is an element of a document having high descriptive value for that document. Let $K_d = keywords(d)$ represent all the keywords of a document;
- A *context* C_x is a document with no specific structure;
- A *dynamic context* ΔC_x is a specialized context that evolves over time;
- An *active context* $C_x A$ is the current specific context;
- A *content* C_n is a specialized document that is to be associated with a context;
- A *corpus* $C = \{d_1, d_2, \dots, d_D\}$ is a collection of D documents;
- A *unified topic model* is a 5-tuple $UTM = \langle V, T, \{\beta_t | t \in T\}, C, \{\theta_d | d \in C\}\rangle$ that describes the set of all the topics T, their distribution over words β_t, the underlying vocabulary V, all the documents C and their distribution over topics θ_d;
- A *category* ψ is a named subset of topics with similar probability distributions over the topic set. The associated topic distribution of the category ψ is denoted by θ_ψ and is organized in a *category taxonomy* Ψ;
- A *contextual relevance* $rel(C_x A, C_n)$ measures the similarity between the context and the content;
- A *dynamic relevance* $rel(\Delta C_x, C_n)$ measures the similarity between the dynamic context and content.

2.1.2.2. The proposed unified thematic model

The proposed model for extracting the thematic structure from a corpus of documents, in a type and structure agnostic manner, relies on the Latent Dirichlet Allocation (LDA) generative model proposed in (Blei D. N., 2003). The LDA model describes a corpus of D documents on which a number of K topics are defined with a β_k topic distribution for topic k. Each document has a θ_d distribution over topics from which a topic $Z_{d,n}$ is sampled for each of the N words in the document. Next, a word $W_{d,n}$ is sampled from $\beta_{Zd,n}$. The model involves two parameters: the α parameter controls the sparsity (Heinrich G. , 2009) of θ while η influences β. A decrease of α leads to a narrowed topic space with higher weights for the selected topics. A decrease of η allows for a limited number of words to be selected for each topic. The observable variables of the models are represented by the words $W_{d,n}$. It is not tractable to directly infer the latent (unobservable) variables (Heinrich G. , 2009) so we considered an approximation technique namely the collapsed Gibbs sampling (Griffiths, 2002).

We started with the solution proposed in (Phan, 2013) and extended it to a parallel Gibbs sampling LDA. We employed an input decomposition technique by dividing the documents analysed during each of the Gibbs iterations in evenly distributed work packages for the parallel processes. The architecture of our solution is basically a pipeline that includes four main processing modules.

The first module, the Keyword Extractor (KE), identifies and extracts the elements of the context with the highest descriptive value. Those elements represent keywords, which outline the significance within the analysed context. This module performs a pre-processing step that prepares the candidates for keyword status by annotating them with the features used in the classification step. The result of this module is a set of n-grams that best describe (summarize) the initial context. They are used as an input by the next module. Formally, KE is described as follows:

$$\mathrm{KE}(C_X A) = K_{C_X A}. \tag{2.1.1}$$

The Topic Identifier (TI) is responsible for associating topic information to the analysed context $(\mathrm{C_X A})$ based on the keywords that describe it $\left(\mathrm{K_{C_x A}}\right)$. The association is accomplished using the TI's underlying topic model. At this point the topic level unification takes place by associating to $\mathrm{C_X A}$ a distribution over topics $\theta_{\mathrm{C_x A}}$. From this point on, all the analysed entities are modelled by a distribution over topics within the unified topic model. Formally, TI is described as follows:

$$\mathrm{TI}\left(\mathrm{K_{C_X A}}\right) = \theta_{\mathrm{C_x A}}. \tag{2.1.2}$$

The Category Combiner (CC) is responsible for computing the similarity between the topic distribution generated by TI ($\theta_{\mathrm{C_x A}}$) and the distribution associated to the managed content ($\theta_{\mathrm{C_n}}$) or dynamic context ($\theta_{\Delta \mathrm{C_x}}$). The main limitation of the topics discovered by TI is anonymity (topics have no semantic information). To overcome this shortcoming we added an abstraction layer on top of the topics called category. Such categories are nodes in a taxonomy having a pre-computed topic distribution. In CC we also analyse the dynamic context that describes the evolution of the interaction based on previously acquired data. The output of this module is a set of advertisements (Υ) with two associated relevance values. One

is computed from the perspective of the active context ($\mathrm{rel}(\mathrm{C_xA}, \mathrm{C_n})$) and the other, from the perspective of the dynamic context ($\mathrm{rel}(\Delta\mathrm{C_x}, \mathrm{C_n})$). Formally, CC is described as follows:

$$\mathrm{CC}\left(\theta_{C_xA}, \theta_{\Delta C_x}\right) = \Upsilon, \tag{2.1.3}$$

where

$$\Upsilon = \left\{ \left\{ \begin{matrix} C_n, \\ \langle rel(C_xA\,, C_n), \rangle \\ rel(\Delta C_x\,, C_n) \end{matrix} \right\} \middle| C_n \in \psi \right\} \tag{2.1.4}$$

With the constant growth of online data, Recommendation Systems face the problem of dealing with huge information spaces. Thus selecting a small representative subset of items, which are not just simply relevant for the user, but at the same time offer an element of novelty, is rather difficult. The diversity measure of the system is focusing to offer users the pleasant surprise of finding an unexpected item of great interest, beyond "obviousness". Overall it is searching for a solution preserving the high quality of retrieved items obtained and offering diversity, therefore serendipity of results.

The *Ranker* (R) is responsible for filtering the output according to the actual performance criteria (Γ). Such criteria can range from relevance to diversity or trustworthiness. In the case of diversity, we aim a low thematic overlap between recommendations while maintaining their relevance to the considered context (whether it is C_xA or ΔC_x). This reduced overlap induces an increased context thematic coverage that is more likely to produce serendipitous recommendations. Formally, R is described as follows:

$$\mathrm{R}(\Upsilon) = \left\{ C_n \middle| C_n \in \mathrm{argmax}_{C_n \in \Upsilon}\, \Gamma \right\}. \tag{2.1.5}$$

2.1.3. MODEL TAILORING ON CONTEXTUAL ADVERTISEMENT

The model we defined is generic and can be applied in various domains. We instantiated our model in the contextual advertisement problem to provide the most suitable advertisement (content) depending on the active context (current Web page) and dynamic context (history of user interaction). The formal mapping of the generic model to the contextual advertisement problem is summarized as follows:

Table 4 Generic model mapping

Generic	**Specific**
Active context	Web page $\mathrm{C_xW}$
Dynamic context	User interaction described by interest I
Content	Advertisement $\mathrm{C_nA}$
Contextual relevance function	$\mathrm{rel}(\mathrm{C_xW}, \mathrm{C_nA})$
Behavioural relevance function	$\mathrm{rel}(\mathrm{I}, \mathrm{C_nA})$

The active context is represented by a web page that can be described by a set of keywords. The advertisements have associated bid phrases that are considered keywords. User history reflects the dynamic context, namely the set of <web page, past ads> pairs the user previously dealt with. Thus, the keywords describing the active context can also reflect the user interests. From the conceptual point of view, we employ a unified technique for the recommendation of

relevant advertisements. Thus, the topic concept describes all the three components of OABM. The advertisement is described by a single, targeted topic. From the active context a static set of topics is extracted. The user is described by a dynamic set of topics to reflect the evolution of interests over time. Let D(θ_e) be a probability distribution over a set of topics describing entity e. Given an entity e we define LDA_e a function from the set of keywords describing the entity e to its probability distribution over topics θ_e as follows:

$$LDA_e: \{\vartheta | \vartheta \in keywords(e)\} \rightarrow D(\theta_e) \tag{2.1.6}$$

We define accordingly the active's context topic distribution as:

$$LDA_{C_xA}: \{\vartheta | \vartheta \in keywords(C_xA)\} \rightarrow D(\theta_{C_xA}) \tag{2.1.7}$$

and the *content* topic distribution as:

$$LDA_{C_n}: \{\vartheta | \vartheta \in keywords(C_n)\} \rightarrow D(\theta_{C_n}) \tag{2.1.8}$$

We claim that $\exists\, t \in T$ s.t. $\text{argmax}_t(\text{p}(t|C_n)) = \{t\}$ where $\text{p}(t|C_n) \in \theta_{C_n}$ hence the content is described by a dominant topic. The dynamic context, representing the user history, is defined based on the hierarchy levels: short (I_s), medium (I_m) and long (I_l) term (Ahmed A. L., 2011). We define the user's overall interest (I) based on a convex combination between the three sub-interests. Let $\kappa \in (0,1)$ and $\text{K} = \kappa + \kappa^2 + \kappa^3$ such that:

$$I = \frac{\kappa}{\text{K}} * I_s + \frac{\kappa^2}{\text{K}} * I_m + \frac{\kappa^3}{\text{K}} * I_l \tag{2.1.9}$$

Let $I_i,\ i \in \{s, m, l\}$ be one of the three sub-interests and $C_i = \{\varepsilon | visit(\varepsilon) \in interval(i)\}$ the set of all accessed contexts during the interval associated with the sub-interest. At this level, we employ the following definition for $I_i, i \in \{s, m, l\}$:

$$LDA_{C_i}: \bigcup_{c \in C_i} keywords(c) \rightarrow D(I_i) \tag{2.1.10}$$

and describe the sub-interest as the distribution over topics of a pseudo-context described by the union of the keywords associated to a context accessed during the sub-interest interval.
In our case-study, the previously mentioned components have domain-specific goals. The Keyword extractor aims to find $D(\theta_e)$ for each $e \in$ {*web page*, *user history*, *advertisement*}. For this purpose, the Feature Extraction (*FE*) sub-module performs the initial pre-processing of the analysed context. We first perform *stop-word removal* and *stemming*. The remaining candidates are enhanced with features that underline their status within the context. We associate to each candidate its occurrences statistic information together with other characteristics dependent on the context's nature like the candidate's localization within the context, its styling information or its inner structure. The *candidates with features* generated by the FE sub-module are persisted in a repository. The Keyword Selection (*KS*) sub-module uses a binary classifier for selecting the keywords from the candidates. The classifier is chosen based on the specific criteria required by Category Assigner (CA). From the business point of view an advertisement recommendation that is out of context is worse than no advertisement at all. Thus, the specific need of our problem is to increase precision as much as possible, by allowing moderate degradation of recall.

The keyword classification process needs to use features that differentiate between the components of the analysed text and filter out bad candidates. Such features can range from statistical descriptors of the occurrences of a word based on both its local and global statistics to localization markers or unique style definitions. Their ultimate goal is to best describe the classification category to which the membership relation is in question.
The Category Combiner module is responsible for computing the triple <web page, user history, advertisement> based on their distribution over topics $\{D(\theta_{C_xA}),\ D(\theta_{C_n}),\ D(I_i)\}$ within the unified topic model. The Category Assigner uses the distribution of topics for the given context and a taxonomy of categories (with their topic distribution) to construct the mapping between a category and a topic. In order to quantify the similarity between two probability distributions we use the Hellinger distance (Nikulin M. , 2011). The module generates a list of candidate categories ordered by their relevance. These categories are further refined by the dynamic context to select a final subset of advertisements that qualify for the next processing step. The dynamic context generation is a process enforced by the Dynamic Behaviour Modeller (DBM). This sub-module aggregates the user information (in the form of interests that is further used to enable behavioural recommendations. The Category-Based Ad Selector analyses ads associated to categories, in a reduced search space due to the category set cardinality reduction yielded by CA and DBM. For each advertisement we further compute two similarity scores (the contextual relevance and the behavioural relevance) based on the Hellinger distance between their probability distributions. Both scores will be further integrated in the ranking module.
The next step aims at ranking the ads based on their relevance that has two components: (1) contextual relevance measured by the <web page, ad> similarity and (2) behavioural relevance measured by <user history, ad> similarity. These two components may be antagonistic by nature as they compete for the same page advertisement slots. The challenge of balancing the two components is addressed by employing a correlation coefficient λ that determines the weights of the two components. The overall recommendation is represented as the weighted sum (convex combination) of the two previously defined relevance functions:

$$\text{similarity}_\lambda(C_xW, I, C_nA) = rel(C_xW\ , C_nA) * \lambda +\ \ rel(I\ , C_nA) *\ (1-\lambda) \qquad (2.1.11)$$

Consequently, the behavioural recommendation maps on $\text{similarity}_0(C_xW, I, C_nA)$ and the contextual recommendation maps on $\text{similarity}_1(C_xW, I, C_nA)$.

2.1.4. EVALUATION AND RESULTS

The first decision to make was to select the classifier(s) to be used. For this, we performed several evaluations with classifier candidates from Weka (Witten, 2011). The experiments were conducted on a dataset with 1333 instances with 75%/25% class distribution. The Percentage Split technique, with 66% for train and 10 repetitions has been considered. The results in Table 5 show that the best performing classifiers are the Multilayer Perceptron (MLP), J48 and a Bagged Predictor with underlying J48 (BJ48). We discarded MLP as candidate due to the large training time required (an order of magnitude compared to the others). The remaining candidates have the same underlying classifier but BJ48 performs additional replications and voting with a minimal improvement of the target measurements.
Thus our final choice is the J48 classifier.

Table 5 Classifier comparison for keyword selection

Name	Correct Classification	Precision	False Positive Rate	F-measure
Bayes Network	89.73	0.96	0.12	0.93
Naïve Bayes	86.44	0.92	0.24	0.91
Multilayer Perceptron	91.17	0.95	0.14	0.94
SMO	86.24	0.90	0.31	0.91
Bagging (J48)	92.04	0.96	0.13	0.94
Decision Table	89.40	0.93	0.22	0.93
J48	91.70	0.95	0.14	0.94
Decision Stump	79.20	1.0	0.0	0.84
AdaBoost M1	87.81	0.93	0.22	0.92
SPegasos	87.70	0.92	0.24	0.92

The largest computational effort appears during the computation of the topic distribution. This process is dependent on the number of words that describe the topic model. We chose to adopt a parallel implementation for this critical area of the flow.

We measured the relative speedup (ΔS) while varying the dimension of input parameters like the number of topics to be discovered (#T), the number of iterations to approach convergence (#I) and the number of analysed documents (#D) for the estimation and inference (inf.) use-cases. For the experimental results covered in Table 6 we used two processing elements. Further investigation showed proportional growth of ΔS as the number of processing elements increases.

We can observe that the growth of a single measure of interest with a controlled increment will generate a proportional growth in both sequential and parallel results by maintaining the relative speedup in a constant range. But when we increase multiple measures of interest with significant increments we observe a spike in the relative speedup hence favouring the parallel implementation.

Table 6 Parallel LDA improvement

Use case	#T	#I	#D	Sequential [s]	Parallel [s]	ΔS
Estimation	30	20	2246	5.52	3.05	1.81
	50	20	2246	8.02	4.40	1.82
	50	40	2246	16.26	8.74	1.86
	100	40	2246	30.39	16.46	1.83
	100	100	2246	77.72	40.47	1.92
	50	40	1123	8.23	4.40	1.87
Inf.	100	100	1123	25.38	14.03	1.81

Two processing elements prove to bring a significant boost for the specific needs.
We further considered a fixed workload scenario where we varied the processing units. Our findings are summarized in Figure 5.

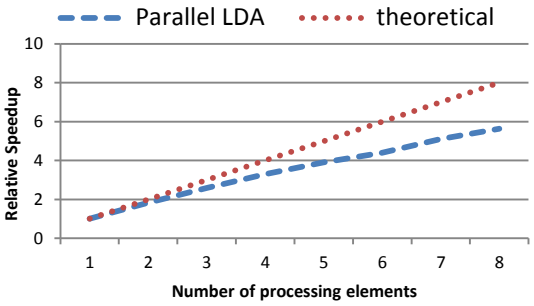


Figure 5 Relative speedup evolution

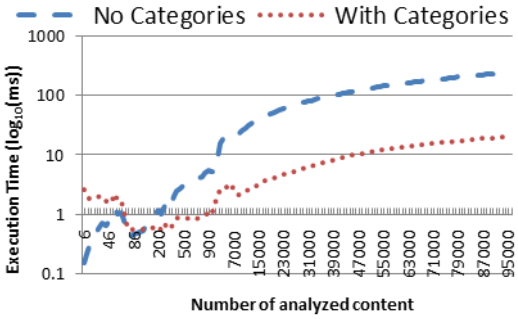


Figure 6 Evolution of category-based selection

We observe 82% efficiency by the time we consider four processing elements and 70% as we get to eight, our available maximum. In a highly parallelized environment, intensive topic model interactions will generate contention on our critical section that makes us slowly converge to our Amdahl limit. For the parallel implementation of LDA we used a shared memory model. Using the technique an average 1.85 relative speed-up for 2 processing elements was obtained.
Another aspect of interest is represented by the benefit introduced by the usage of categories as an additional abstraction layer above topics. Figure 6 shows the evolution of execution time as the number of analysed contexts grows both with and without the usage of categories. This behaviour appears in the CA sub-module of the CC.
Our category taxonomy has 100 nodes organized on 6 levels. We can see that even if the category based approach starts with a default overhead, as the number of advertisements grows, the two curves will intersect when the total number of advertisements equals the number of categories combined with the number of ads in the top categories and from that point on, their growth patterns differ with almost a *decade*.

Another aspect we considered is the evaluation of the provided recommendations. Since the nature of the problem is highly subjective and there are no annotated benchmark datasets, we chose to employ an end user evaluation of the recommended advertisements. A group of users were asked to assess on a 1-10 scaling rate the similarity of the recommended content with the designated context. We present the results in Table 7, showing that the user feedback correlates well with the results of the Hellinger distance.

Table 7 User evaluation compared with thematic similarity

Context	Content (ad)	User Average Points (UAP)	Hellinger Distance (HD)	HD per UAP
C27	A26	8	0.63	0.079
	A910	9	0.57	0.064
	A867	2	0.88	0.441
C42	A283	3	0.93	0.309
	A736	6	0.77	0.130
	A882	7.5	0.70	0.099
C54	A801	2	0.86	0.427
	A884	6	0.77	0.127
	A128	2	0.88	0.438

Exploring further the influence of the correlation coefficient λ on the recommendations we asses the marginal cases of fully contextual (λ=1) and fully behavioural (λ=0) recommendations as well as the balanced combined (λ=0.5) cases. Let's consider the context (both user behaviour and web page) having their top three topics illustrated in Table 8.

Table 8 Top three topics of analysed context (user and web page)

	Topic1		Topic2		Topic3		Combined Coverage
User (U1)	T21	25%	T22	17%	T44	50%	92%
Web Page (C27)	T5	35%	T27	18%	T48	15%	68%

These topics cover, in different proportions, the context but their combined coverage is sometimes enough to describe them. We consider a set of recommended advertisements for which we compute the degree of coverage and the combined similarity using different versions of the correlation coefficient. We conclude that there is a correlation between the low values of the Hellinger distance and the associated coverage from both perspectives. We also considered a manual user evaluation for exploring the correlation with the Hellinger distance. The relation between the Hellinger distance and the user's evaluation is represented in the curve depicted in Figure 7.

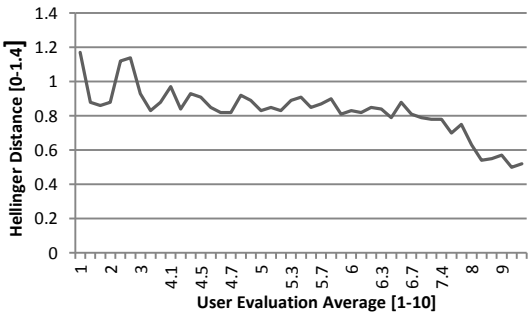


Figure 7 Hellinger evolution with user evaluation

At the lowest end of the evaluation interval (ranging between 0 and 4) one can observe a strong variation between the actual distance and the score. But at this level is difficult to

assess the correlation because users see content within this band from different perspectives. The important aspect is that content in this range is not desirable.
On the other hand, at the other extremity of the interval one can observe an evolution of the distance to ever-decreasing values. The area of confusion is between 4 and 7. Here, the distance varies with small increments making it harder to discriminate. We concluded that Hellinger values above 0.9 have a higher chance of being bad content and values below 0.7 of being good content. If we consider that this measure is theoretically bound between 0 and $\sqrt{2}$ this results seems promising.

2.1.5. CONCLUSIONS

In this work we proposed a model capable of providing context-sensitive content based on the similarity relation between the analysed context and the recommended content. The similarity is measured at thematic level by mapping both the context and the content to a common reference system, the proposed unified thematic model. In order to obtain topic information, we analyse the underlying thematic structure induced by the keywords of the considered entities (contexts, both static and dynamic, and content) using a parallel topic extraction approach. The content search space is reduced using the category abstraction and the quality is measured based on relevance and diversity.
We applied the model on the contextual advertising problem considering two components of the context: the active context is the current web page, the dynamic context is the user interaction within the system and the content is the advertisement.

2.2 Supervised, cross-language opinion mining approaches

2.2.1. INTRODUCTION

Sentiment analysis is one of the hottest topics in the information retrieval field since the availability of huge amounts of user generated content. Users are now able to express themselves on any kind of topic (products/services, politics, etc.) in various types of media such as articles, blogs, comments, reviews, tweets, Facebook postings etc. The resulted content can reveal useful information not only for individual users that are searching for a certain product or service but also for companies to understand the global opinion of their customers related to their products or to tune their campaigns according to the user feedback. For example, if a new product receives several bad reviews on a certain aspect, the company might react timely to improve that aspect before the sales of that product drop significantly.
The main issue of the sentiment analysis field is to correctly identify the semantic of the user generated content in terms of opinions out of natural language text that might be grammatically incorrect or might contain special symbols like hashtags, emoticons etc. Moreover, not any user generated content might express opinions or sentiments so the first challenge is to discriminate between opinion bearing and non-opinion bearing content. A first discrimination would make sense between objective and subjective content. Objective content represents facts that normally do not involve any sentiment. Even in case of subjective content, it might not involve sentiments or opinions, just user perceptions. Therefore, identifying correctly user expressed opinions is not a trivial problem. A first step towards opinion identification is to understand what we are looking for. In other words we should know what the components of an opinion are in order to recognize them in texts. A widely adopted structure for opinions was proposed in (Hu, 2004) and (Liu, 2010) as a quintuple <e,a,s,h,t> where e represents the addressed entity/target, a represents a particular

aspect of the entity e the opinion is about, s is the actual sentiment on the aspect a, h is the holder of the opinion and t is the time the opinion was expressed.
Obviously, these opinion components are seldom explicit and clear in the text. In many cases we have to deal with implicit components such as implicit aspects. For example in the sentence "The printer is not expensive but the ink cartridge is" the involved aspect of the printer is the price but it is not explicitly mentioned. Other implicit opinions are related to pronouns that represent previously mentioned entities that can be targets or holders, or comparative opinions. In this case the opinion does not provide an absolute evaluation of the target/aspect but a relative one to another product.
In our work so far we addressed certain facets of the sentiment analysis problems and obtained relevant results.

2.2.2. PROBLEM OVERVIEW

Opinion mining is a challenging task that involves various types of problems and for each problem several approaches have been reported in literature. We are briefly mentioning some of them in the next sections.

2.2.2.1. Problem facets

The main task of sentiment analysis is the accurate identification of opinions with all the components defined above from different types of raw input text. This task is not trivial since it involves not only the identification of the sentiment polarity but also associating it to the correct entity or aspect.
One facet of the problem is the **granularity level**. Research results found in literature are addressing three granularity levels: document, sentence and aspect level. We present in next section existing approaches for each of these levels. Another facet of the problem is the **language issue**. Most of the approaches are addressing documents written in English. Several high quality tools and data resources have been developed for the English language so far. There are some approaches that handle other languages such as Chinese, Japanese, Arabic etc. but there is no proven solution to be language independent or even to perform similar in several languages. Moreover, in the framework of the same language, even English, we can distinguish between correctly written documents (e.g. articles, news) and documents that contain errors, special symbols, hashtags, emoticons (e.g. reviews, tweets, Facebook posts/comments etc.). The latter require additional processing besides traditional Natural Language Processing (NLP) in order to handle the special content. In terms of sentiment polarity identification several nuances can be considered: the first discrimination that can be made is between objective and subjective content. As mentioned before objective content basically represents facts but also facts can include sentiments. For example the phrase "Israel's bombardment has left hundreds of thousands in strip without power, water or access to basic medical supplies." states a fact but it definitely involves a negative sentiment. On the other hand, subjective sentences might express user beliefs that are not necessarily involving a sentiment, such as "I think the dress was blue". So, discriminating between subjective and objective statements is not similar to identifying sentiment bearing statements. Another challenge would relate to the **number of classes** to which the sentiments can be mapped. The most straightforward and common approach is to classify sentiments as positive or negative with a possible additional neutral class. A more refined approach would consider several classes such as the six types of primary emotions defined in (Parrott, 2001): love, joy, surprise, anger, sadness, and fear. These can be subdivided in secondary and tertiary emotions, each having different intensities.

2.2.2.2. Relevant approaches

We presented here only a brief overview of the problem facets that have been addressed by different research approaches. There is obviously no general solution to cover all of them. A possible taxonomy of the most common approaches would include supervised and unsupervised solutions, at all three levels of granularity (i.e. document/sentence/aspect). From the domain dependence perspective, we might classify the approaches in domain dependent, cross-domain and domain independent. Most supervised techniques are domain-dependent since they rely on training datasets belonging to a domain so that a classifier trained on the given dataset will perform good on data from the same domain, but usually poorer on data from a different domain. Therefore, different approaches to reuse the classification models learned from one domain to different domains were proposed and will be discussed in the next section. Approaches that do not rely on training data, such as rule-based approaches, are more likely to be domain independent.

We can also consider the language as a structuring criterion therefore classifying the approaches as one-language, cross-language or multi-language approaches. Similarly to the domain dependence issue, most of the existing approaches address documents written in one language, namely English. Also, several high-quality linguistic processing resources were developed for the English language. Another research direction is concerned with adapting/transferring the models built for English documents to other languages or even to develop models that can be applied to a family of languages, given the performance of automated translation tools and the existing linguistic resources for English.

2.2.2.3. Practical applications

There is so much information to be derived beyond opinions. At the business decisions level, the individual reviews or customer feedback cannot be handled. Therefore the overall opinion that is statistically relevant has to be derived out of all the individual opinions. Opinion summarization can be considered at several levels: aggregating the overall opinion about an entity out of the opinions expressed in reviews, or just aggregating the opinion on an entity out of the opinions expressed on different aspects of that entity.

Moreover, sometimes opinions are not expressed in absolute terms, but relative to other entities (e.g. similar products of the competition etc.). In this case the aim is to identify the two entities that are compared, their common aspects and the preference order of the holder.

Having a huge amount of identified opinions opens new perspectives such as time analysis by identifying opinion trends. Moving on the social computing path we can also identify opinion-driven communities and their changes over time.

Another interesting and challenging task is the identification of opinion spam. Dishonest opinions meant to mislead the readers should be identified and excluded from the opinion analysis.

2.2.3 SUPERVISED, DOMAIN-INDEPENDENT APPROACH FOR ENGLISH

Our first endeavour was concerned with developing a solution that is as domain independent as possible, given that it is based on a set of training data. At this point we are not experimenting with other languages than English, trying to get the best performance that we can for English documents. We propose an approach to document-level sentiment polarity identification that leverages a combination of three meta-feature classes (Hasna O.L., 2014). The novelty of the approach relies on the feature-vector characteristics: as the classification instances are characterised via meta-features, the model gains in generality being domain cvasi-independent.

We utilise the following three classes of meta-features:

- Part-of-speech patterns representing syntactic constructs with increased sentiment promise;
- Polarity histograms grouping the words of a document in buckets based on their sentiment polarity;
- Sentiment lexicons representing a proven collection of words annotated with polarity information.

We incorporate the sentiment polarity identifier in a context-sensitive recommendation workflow. The aim of this use-case is to associate to an input collection of unstructured documents (context) the most appropriate content. We define the content as a document already structured and tagged. Documents are tagged with their sentiment polarity as a result of the classification based on meta-features. Structural information is associated with respect to a thematic reference system.

In the following we formally define the basic notions used throughout the article.

- The *sentiment polarity* of an entity sp_e describes an ordered distribution over orientations: *Positive* (ρ), *Negative* (η), *Objective* (o) such that $sp_e=\langle\rho,\eta,o\rangle$ where $\rho+\eta+o=1$;
- The *sentiment orientation* of an entity $so_e\in\{0,+,-\}$ and is described as $so_e=\text{argmax}(x)$, $x\in sp_e$
- A *word* w is the basic unit of discrete data (Blei D. N., 2003), defined to be an element of a vocabulary V. A *word* can have a *sentiment polarity* sp_w;
- A *topic* $\beta_t=\{\ P(w|t)\ |\ w\in V\}$ is a probability distribution over a finite vocabulary of words where $\sum_{i\in\beta_t} i = 1$;
- A *document* $d=W_d=\{w_1,w_2,...,w_n\}$ is a sequence of words in any language;
- A *keyword* k is a word of a document having high descriptive value. Let K_d represent all the keywords of document *d*;
- A *context* is defined by a collection of weighted documents of arbitrary structure; $D_{context}=\{\langle d,p_d\rangle|d\in Context\}$, where $\sum_{d\in D_{context}} p_d = 1$;
- A *content* is defined by a collection of labelled documents which have a well-defined structure. Each document has a *sentiment polarity* sp_d and an associated distribution over topics $\theta_d=\{P(t|d)\ |\ t\in T\}$ where $\sum_{i\in\theta_d} i = 1$; $D_{content}=\{\langle d,sp_d,\theta d\rangle|d\in Content\}$

2.2.3.1 Meta-features for sentiment classification

Our approach aims to achieve domain-independence by inferring sentiment polarity using meta-features. The goal is to generalize the members of the feature vector so as not to bind them to the characteristics of a specific domain. We employ three main classes of features: sentiment lexicons, part-of-speech patterns and polarity histograms.

A **sentiment lexicon** (SL) is represented by a collection of words annotated with their sentiment polarity. Formally it can be described as follows:

$$SL_V = \{\langle w,\rangle|w\in V\} \tag{2.2.1}$$

We also define the basic operations (union, intersection and difference) on sentiment lexicons. An operation applied on a SL is equivalent with applying that operation on their associated vocabulary. In case of vocabulary overlaps, collisions are resolved by selecting the sentiment polarity of the SL with the highest priority. Thus, we define the following:

$$SL_{V_i} \cup SL_{V_j} = \left\{ \begin{pmatrix} w, \\ \langle sp_{w,x} \rangle \end{pmatrix} \middle| \begin{matrix} w \in V_i \bigcup V_j \wedge \\ x = argmax\left(P_{SL_{V_i}}, P_{SL_{V_j}}\right) \end{matrix} \right\} \quad (2.2.2)$$

$$SL_{V_i} \cap SL_{V_j} = \left\{ \begin{pmatrix} w, \\ \langle sp_{w,x} \rangle \end{pmatrix} \middle| \begin{matrix} w \in V_i \bigcap V_j \wedge \\ x = argmax\left(P_{SL_{V_i}}, P_{SL_{V_j}}\right) \end{matrix} \right\} \quad (2.2.3)$$

$$SL_{V_i} \setminus SL_{V_j} = \left\{ \begin{pmatrix} w, \\ \langle sp_{w,SL_{V_i}} \rangle \end{pmatrix} \middle| w \in V_i \setminus V_j \right\} \quad (2.2.4)$$

Our approach uses as lexicon a combination between two collections commonly used in literature. The first lexicon is proposed in (Hu, 2004) and represents a list of positive and negative sentiment words for English. Depending on orientation, we associated to each word in this lexicon one of the following polarities: $sp_w^+ = \langle 0.9, 0.05, 0.05 \rangle$ or $sp_w^- = \langle 0.05, 0.9, 0.05 \rangle$. We denote this sentiment lexicon as SL_{HuLiu}.

The second resource we leverage is *SentiWordNet* (SWN) (Baccianella, 2010). SWN is the result of an automatic annotation of all *WordNet* synsets (*ss*). As a result, each synset receives a positive and a negative polarity. SWN uses the WordNet structure which groups similar meanings of different words in a synset. A word can be part of multiple synsets by exhibiting a different *sense*. So a word can have different sentiment polarities based on the sense it plays in the analysed document. Let $sense_w$ be the set of synsets associated to a word in SWN.

In order to build a sentiment lexicon associated to SWN (SL_{SWN}) we need to associate a single polarity to each word. Words might be associated with multiple synsets ($|sensew| \geq 1$). This is why we define the multi-synset fall-back schema (MSFB). It associate to each word either their synset's polarity if the word is part of a single synset or the polarity of the synset that maximizes the absolute difference between their positive (ρ) and negative (η) polarity. This is why we define:

$$SL_{SWN}=\{\langle w,MSFB(w)\rangle | w \in V_{SWN}\} \quad (2.2.5)$$

$$\text{MSFB}(w) = argmax_{ss_i \in sense_w} \left|\rho_{sp_{ss_i}} - \eta_{sp_{ss_i}}\right| \quad (2.2.6)$$

Since many of the synsets are objective we choose to define d_{SWN} as the subset of SWN with synsets that have a distinguishable positive or negative polarity. Thus, we define

$$dSWN = \left\{ ss \in SWN \middle| \begin{matrix} sp_{ss} = \langle \rho, \eta, o \rangle \wedge \\ \rho \neq \eta \wedge \\ (\rho > o \ \vee \eta > o) \end{matrix} \right\} \quad (2.2.7)$$

This reduces the number of synsets and helps us underline the sentiment baring words. We call such sentiment lexicons *strongly distinguishable* (SL_{HuLiu} is also strongly distinguishable). A sentiment lexicon SLV is *strongly distinguishable* if, for any $w \in V$ the condition (2.2.7) stands.

Applying MSFB, we build SL_{dSWN} as the sentiment lexicon associated to *dSWN*. An interesting consequence is that the percentage of words with a single synset grows. Furthermore, we are interested in analysing the vocabulary overlap between SL_{dSWN} and SL_{HuLiu}. With the help of relations (2.2.2), (2.2.3) and (2.2.4) we can further refine lexicon combinations.

We leverage sentiment lexicons as domain-independent meta-features. They represent a fixed set of words that are to be searched in document instances. We measure a Boolean meta-feature (i.e. whether or not an element of the lexicon appears in the document instance). The feature vector associated to a SLV is described as follows:

$$fSL_V(\mathrm{d}) = \left\langle i = \begin{cases} 1, w \in W_d \\ 0, w \notin W_d \end{cases} \middle| w \in V \right\rangle \quad (2.2.8)$$

Another interesting aspect of sentiment lexicons is negation. Any word might appear in a negated context which inverses its sentiment polarity:

$$(sp_w) = \{\langle \rho', ', o' \rangle | \rho' = \eta \wedge \eta' = \rho \wedge o' = o\} \quad (2.2.9)$$

In the rest of the paper, we will refer to combinations between sentiment lexicons based on the applied set of operations (e.g. the union between SL_{HuLiu} and SL_{dSWN} becomes $SL_{HuLiu\_\cup\_dSWN}$).

Part-of-speech patterns (($w_{1,2}$)) represent a specialized combination of words tagged with POS information. In (Turney, 2002) five such two-word patterns are used to extract bigrams with increased sentimental promise. Table 9 describes the part-of-speech involved in them.

Table 9 Turney POS patterns

i	First Word POS (w_1)	Second Word POS (w_2)	Third Word (not extracted)
1	Adjective	Noun	Anything
2	Adverb	Adjective	Not Noun
3	Adjective	Adjective	Not Noun
4	Noun	Adjective	Not Noun
5	Adverb	Verb	Anything

A bigram ($\langle w_{j,+1} \rangle$) will match the i[th] part-of-speech pattern ($POSp_i$) if the following relation stands:

$$POSp_i(w_j, w_{j+1}) = \langle pos_{w_j} pos_{w_{j+1}} \rangle \equiv POSp_i \quad (2.2.10)$$

We propose the usage of these patterns as meta-features in two instantiations (i.e. one for the positive orientation ($POSp_{+}$) and one for the negative ($POSp_{-}$)) thus generating 10 meta-features.

The polarity of a POS pattern is computed based on a linear combination between the sentiment polarities of the two words that are part of the pattern. Instances with a distinguished positive polarity count for $POSp_{+}$. If the negative polarity is distinguished it will count for $POSp_{-}$. We describe the relation as follows:

$$sp_{w_j, w_{j+1}} = \omega * sp_{w_j} + (1 - \omega) * sp_{w_{j+1}} \quad (2.2.11)$$

where ω is an experimentally computed coefficient and sp_{w_j} and $sp_{w_{j+1}}$ are retrieved from a sentiment lexicon. We've determined that a good ω would be 0.5 for $POSp_2$ and $POSp_3$. For

the other three, we associate 0.8 for the adjective or adverb. We treat negation for $POSp$ by considering ($sp_{w_j,w_{j+1}}$) as the pattern's polarity.

The count of an individual $POSp$ instance in a document is described as follows:

$$cnt_{POSp}(d,i,o) = \left| \left\{ j \left| \begin{matrix} w_j \in W_d \\ POSp_i(w_j, w_{j+1}) \\ so_{w_j,w_{j+1}} = o \end{matrix} \right. \right\} \right| \quad (2.2.12)$$

where w_j is a word from d where the ith pattern is instantiated with the proper sentiment orientation. For each document, we compute the part-of-speech patterns feature vector as a 10-tuple described by the following relation:

$$fPOSp(d) = \langle cnt_{POSp}(d,i,o) \,|i = \overline{1,5}, o \in \{+,-\} \rangle \quad (2.2.13)$$

The third set of meta-features is defined by **polarity histograms**. Polarity Histograms are a measure of the degree to which a document contains words of different sentiment polarities. We analyse words from the document that exhibit sentimental promise based on the polarity lexicon. We group them in buckets of size Δ on a two-dimensional lattice. The actual bucket size depends on the polarity values reported by the sentiment lexicon.

The diameter of a disc in Figure 8 and Figure 9 represent the number of words that have a positive sentiment polarity within $[x, x + \Delta_x)$ and a negative polarity within $[y, y + \Delta_y)$.

Figure 8 depicts the polarity histogram of a positive document. It uses $SL_{HuLiu\_U\_dSWN}$ as sentiment lexicon. There are no words in buckets bellow 0.5 because the lexicon is strongly distinguishable. In Figure 9, we represent the polarity histogram resulted from processing a negative document using the $SL_{HuLiu\_U\_SWN}$ lexicon. The lexicon used for this document lacks the distinguishability property.

We adopt polarity histograms as the third set of meta-features as they capture the overall polarity information of a document, normalized to a given lexicon. In the polarity context, negations have the same impact as for POS patterns. Thus, we inverse the sentiment polarity for a word if it occurs negated in the document.

The size of an individual bucket is measured as follows:

$$cnt_{PH}(bucket, d) = \left| \left\{ j \left| w_j \in W_d \wedge sp_{w_j} \in bucket \right. \right\} \right| \quad (2.2.14)$$

where w_j is a word from d and its sentiment polarity sp_{w_j} falls within the $bucket$.

The polarity histogram feature vector that corresponds to a document d is described as follows:

$$fPH(d) = \langle cnt_{PH}(b,d) \,|b \in Buckets \rangle \quad (2.2.15)$$

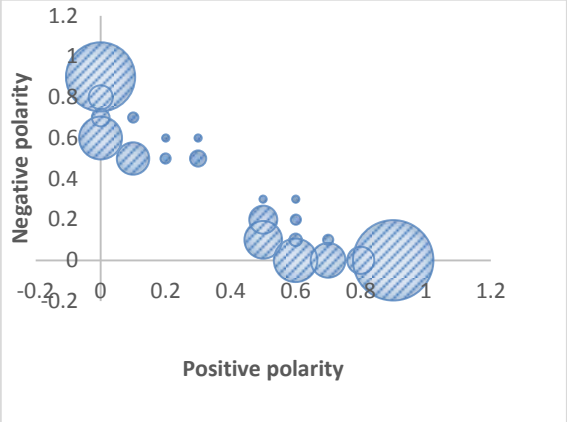


Figure 8 Polarity histogram for a document with positive sentiment orientation using $SLHuLiu\_\cup\_dSWN$

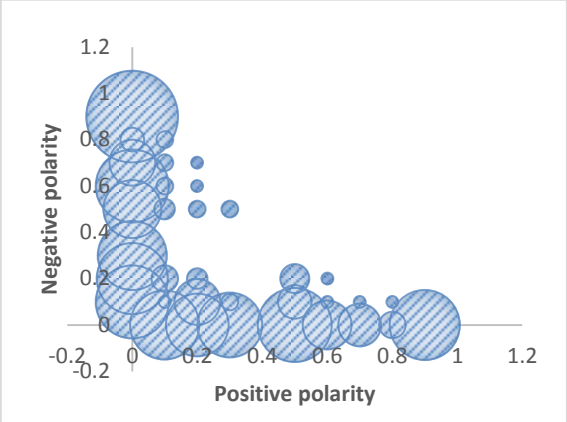


Figure 9 Polarity histogram for a document with negative sentiment orientation using $SLHuLiu\_\cup\_SWN$

2.2.3.2 The Sentiment classifier

In our processing flow we employ a Sentiment Polarity Identifier (SPI) that is responsible for associating a sentiment polarity (spd) to a document (d).

The first step of the polarity identification process is deciding whether or not the input document is subjective. This helps us filter out objective documents (i.e. that lack polarity). To this purpose, we adapt the work of (Lin, He, Everson, & Rüger, 2012) for the task of subjectivity detection. They propose the Joint Sentiment-Topic model (JST) as an extension to LDA. The generative process for JST follows three stages. One samples a sentiment label from a per-document sentiment distribution. Based on the sampled sentiment label, one draws a topic from the sentiment-associated topic distribution. Finally one chooses a word from the per-corpus word distribution conditioned on both the sampled topic and sentiment label. In order to infer the latent thematic structure, they use a collapsed Gibbs sampling approximation technique. We built upon their approach a binary classifier. It is responsible for analysing the distribution over sentiment labels of an input document.

Subjective documents are further analysed with respect to the three classes of meta-features. To better measure the sentiment orientation of a word, we start with negation detection for which a naïve approach is applied, that searches for words that are part of a negation lexicon. So far only explicit negations are considered (words negated by not, don't and similar). Each time such a word is detected its determined word is marked as negated.

Next, the polarity tagging step attaches to each processed word its associated sentiment polarity using the configured sentiment lexicon. The polarity of the document is aggregated from the document's words which belong to the lexicon as well. The sentiment polarity of a word is inversed if it is preceded by a negation. Polarity enriched words are the input for all three meta-feature extractors.

At this point we start collecting instances of our three meta-feature classes. The POS patterns extractor collects instances of the 10 POS patterns. Then we apply the polarity histogram extractor which starts filling in the defined buckets based on the individual polarity of words in the analysed document. Finally, we apply the polarity words extractor that selects the words that are part of the sentiment lexicon. At this level, we treat negation by doubling the size of the lexicon's vocabulary (each word gets negated). The feature vector of a document (fv(d)) used by the polarity classifier is formally described as

$$\text{fv}(d) = fPOSp(d) \cup fPH(d) \cup fSL(d) \tag{2.2.16}$$

It contains the following meta-features:

- 10 meta-features whose values represent the number of part-of-speech pattern instances of each type found in the document;
- For each polarity histogram bucket, the number of words with sp_w within that bucket;
- For each word in the polarity lexicon, a Boolean marker describing its membership in W_d.

2.2.3.3 Experiments and Results

We considered several perspectives to evaluate the quality of our results and to fine-tune the classification models accordingly.

First, we analyse the degree to which the subjectivity classifier manages to point out objectivity. For this purpose we evaluate the binary classifier on the subjectivity dataset introduced in (Pang, 2004). It consists of 5000 subjective and 5000 objective processed sentences extracted from snippets of movie reviews and plot summaries. We vary the number of Gibbs sampling iterations for model estimation (ME) and inference (I) and measure average weighted precision (awp) and recall (awr) with their standard deviation (σ). Furthermore, we want to reduce the number of false negative for the objective class (fnr_o) thus limiting the number of objective documents that get to polarity identification. We evaluated each custom classifier configuration using the 10-fold cross-validation mechanism. Based on the results we obtained we choose the ME500_I50 configuration. At the cost of an increased variability, it manages to maximize awp and awr while reducing fnr_o. This is an acceptable trade-off for our task.

Second, we were interested in the effect of applying the distinguishability property on $SLSWN$. We analyse the dimensionality reduction induced by this property with a focus on the proportion of words that end up being associated with a single synset. In Table 10 we show the number of synsets (#syn) and words (#w) in both sentiment lexicons. SL_{dSWN} has 94.1% less words and 95.1% less synsets than SL_{SWN}. Furthermore, we measure the distribution of words (SynPerWord) that are associated to a single synset, 2 or 3 synsets or more. Moreover, for 85.4% of the words from SL_{dSWN} a unique sentiment polarity can be associated from their corresponding synset. This means that we rely on the multi-synset fall-back schema (relation (2.2.6)) 4 times less than for SL_{SWN}.

Table 10 Comparison between SWN and dSWN

SL_V	#syn	#w	SynPerWord		
			1	2or3	more
SWN	117659	147306	.401	.124	.475
dSWN	5736	8548	.854	.134	.012

The third aspect that we analyse is the best configuration for the feature vector. Based on preliminary results for initial evaluations on different classifiers (NB, SVM and C4.5), we restricted the evaluations to the Naïve Bayes (implementation available in Weka) configured to use a kernel separator (Witten, 2011). While evaluating the configuration of the classifier we use the version 2.0 of the Movie Review Dataset (MR) first introduced by (Pang, 2004). It consists of 1000 negative and 1000 positive movie reviews crawled from the IMDB movie archive. The average document length is 30 sentences.

For validation, we randomized the dataset, hide 10% for evaluation and split the remaining 90% into 10 folds. The classifier is trained 10 times on 9 different folds (81% of the corpus), tested on 1 fold and evaluated against the hidden 10%. We repeat this process with multiple random seeds.

The evaluations were performed on the data set with different features combinations. The first set of features represents the elements of a sentiment lexicon. The candidate lexicons are

SL_{dSWN}, SL_{HuLiu} and the lexicons obtained by applying the basic set operations on the two (denoted correspondingly in Table 11). By structurally analysing the lexicons, we measured the number of words (#w) in each and the distribution of positive (#pw) and negative (#nw) word sentiment polarities. In Table 11 we compare the lexicon candidates with respect to their vocabulary size. It's interesting to note for all lexicons the distribution of negative words is greater than the distribution of words with a positive sentiment orientation.

Table 11 Sentiment lexicon operations comparison

SL_V	#w	#pw	#nw
$HuLiu$	6786	.295	.705
$dSWN$	8548	.359	.641
$HuLiu\_ \cup \_dSWN$	13080	.336	.664
$HuLiu\_ \cap \_dSWN$	2254	.302	.698
$HuLiu\_ \setminus \_dSWN$	4532	.291	.708
$dSWN\_ \setminus \_HuLiu$	6294	.380	.620

Using only such features while classifying instances from MR we measured the average weighted precision and recall for the positive and negative classes together with their standard deviation. The results in Table 11 suggest that the best sentiment lexicon choice is the union between the two lexicons. Furthermore, the vocabulary intersection sub-set is more valuable than any of the sub-sets specific to one of them.

Table 12 Lexicons evaluation

SL_V	awp	σ_{awp}	awr	σ_{awr}
$HuLiu$	.755	.026	.746	.026
$sdSWN$	.727	.016	.724	.016
$\boldsymbol{HuLiu\_ \cup \_dSWN}$	**.767**	**.029**	**.758**	**.029**
$HuLiu\_ \cap \_dSWN$	.711	.013	.708	.014
$HuLiu\_ \setminus \_dSWN$	.668	.025	.664	.026
$dSWN\_ \setminus \_HuLiu$	.629	.021	.627	.018

Next the evaluation is seeking for finding the best feature set, using as candidates our three meta-feature categories, *part-of-speech patterns* (POSP), *polarity histograms* (PH) and the *sentiment lexicon* (SL). Seven experiments were performed each considering a different combination of the three categories of feature vector candidates. For polarity histograms we set the bucket sizes Δ_x and Δ_y to 0.1. The results in Table 13 show that each of the three meta-feature classes brings an incremental improvement. The biggest impact is brought by the sentiment lexicon. The best feature vector contains the combination between part-of-speech patterns, polarity histograms and the sentiment lexicon.

Table 13: Feature vector composition from meta-features.

Configuration	awp	σ_{awp}	awr	σ_{awr}
POSP	.642	.027	.637	.031
PH	.640	.029	.624	.028
SL	**.767**	**.029**	**.758**	**.029**
SL + POSP	.816	.014	.814	.014
SL + PH	.825	.016	.820	.016
PH + POSP	.671	.027	.664	.033
SL + POSP + PH	.841	.015	.829	.016

To asses domain independence we have tested the feature vector configuration on other domains. Proposed by (Blitzer, Dredze, & Pereira, 2007) the Multi-Domain Sentiment

(MDS) dataset is a collection of Amazon reviews from multiple domains. It consists of 26 domains with labelled positive and negative reviews. We've considered in our experiment 14 domains that have more than 800 positive and 800 negative labelled reviews. In literature, the initial 4 domains are extensively used for evaluation. They cover the Book (B), DVD (D), Kitchen (K) and Electronics (E) and have 1000 positive and 1000 negative reviews. For this experiment we also measure classification accuracy (*acc*) because this is the metric used for comparison in other studies using the MDS-4. Table 14 reports the results for 10 random seeds with two outliers excluded (min & max *awp*) on both MR and the 15 domains of MDS.

Table 14: In-domain verification using multiple domains.

Dataset	awp	σ_{awp}	awr	σ_{awr}	acc
MR	.841	.015	.829	.016	82.87
Book (b)	.740	.025	.712	.029	71.89
DVD (d)	.807	.023	.800	.026	80.03
Electro (e)	.801	.025	.796	.024	79.59
Kitchen (k)	.849	.019	.847	.018	84.69
Apparel (a)	.853	.019	.851	.019	85.12
Baby (ba)	.837	.022	.836	.021	83.62
Camera (c)	.855	.022	.851	.023	85.13
Health (h)	.810	.025	.808	.024	80.86
Magazine (m)	.857	.018	.852	.019	85.26
Music (mu)	.792	.023	.789	.024	78.99
Software (s)	.825	.029	.818	.032	81.87
Sports (sp)	.819	.026	.816	.027	81.67
Toys (t)	.829	.021	.825	.023	82.57
Video (v)	.762	.038	.741	.047	74.10
AVERAGE	.818	.040	.811	.046	81.21

We compare our approach with other studies that leveraged the same datasets for in-domain classification (training and validation on the same domain). In Table 15 we compare against the results of in-domain testing of (Lin, He, Everson, & Rüger, 2012), (Raaijmakers & Kraaij, 2010) and (Blitzer, Dredze, & Pereira, 2007) and against the results obtained with NB from (Xia, Zong, & Li, 2011).

Table 15: Accuracy comparison with literature.

	MR	B	D	E	K
Lin2012	76.6	70.8	72.5	75	72.1
Xia2011PoS	82.7	76.7	78.85	81.75	82.4
Xia2011WR	85.80	81.2	81.7	84.15	87.5
RK2010	N/A	78.8	82.3	86.5	88.8
Bli2007	N/A	80.4	82.4	84.4	87.7
Proposed	82.87	71.18	80.03	79.59	84.69

2.2.4 SUPERVISED OPINION MINING FOR ROMANIAN

Our next concern was to investigate how to transfer our classification models that rely on language specific processing (POS-tagging, entity extraction rules etc.) and resources (Stanford Parser) to perform acceptable for Romanian documents. In this respect we addressed both the aspect level opinion entity and polarity identification and the document level sentiment classification (Anisiei, 2014), (Russu, 2014).

The first task employs a rule-based approach while the text classification employs a supervised one. The first process operates at the aspect level and involves the extraction of opinion words from Romanian natural language texts, based on the relationships between

words. We have identified 7 relevant relationships presented in the Opinion Identification section. Further, the process employs a polarity identification task on the extracted opinions.

2.2.4.1. Opinion Identification

In order to identify the opinion we employ a flow that involves two tasks: the opinion extraction and the polarity identification. The natural language unlabelled input text is fed into the dependency parser that will provide the PartOfSpeech (POS) and lemma of each word. To identify the opinions we employ a set of relevant relationships between the words in the parsed document. Considering the Romanian language, words that are in some grammatical relationship are identified, and they are denoted by the dependent and governor. Considering w1 and w2 as words, R represents the syntactic relationship between w1 and w2. Generally, the possible relations are

$$R=\{a.adj., coord., aux.,c.c.m., n.pred., sbj., neg.\ part.\} \quad (2.2.17)$$

The possible POS for the two involved words are

$$POS(w_{i=1,2})=\{N, adj., det.,adv.,cnj.,vb.\} \quad (2.2.18)$$

We identified the following rules that are likely to represent opinions in documents:

$$R1: R = \{a.adj.\}, POS(w_1)=\{adj.,det.\}, POS(w_2)= \{N\} \quad (2.2.19)$$

In this case the governor(w_1) is either an adjective or a determiner. The dependent(w_2) in this case is always represented by a noun. E.g. *Astazi este ziua mea și am primit un tort delicios. (Today is my birthday and I received a delicious cake.)*
In this example the governor (w_1) is "delicios" (delicious) and the dependent word (w_2) is "tort" (cake).

$$R2: R = \{coord\}, POS(w_1)=\{adj.,adv.\}, POS(w_2)= \{cnj\} \quad (2.2.20)$$

In this case the governor(w_1) is either an adjective or an adverb while the dependent(w_2) is a conjunction. E.g. *Astazi este ziua mea și am primit un tort frumos și delicios. (Today is my birthday and I received a beautiful and delicious cake.)*
In this example the governors are "frumos" (beautiful) and "delicios" (delicious) and the dependent word is "tort" (cake).

$$R3: R = \{aux\}, POS(w_1)=\{N.\}, POS(w_2)= \{adj\} \quad (2.2.21)$$

E.g. *Tânărul mergea atent de-a lungul zidurilor, scrutând, acolo unde lumina slabă a felinarelor îngăduia, numerele caselor. (The young boy was walking carefully along the walls, peering, where the dim light of lanterns permits, the house numbers.).*
In this example the aux. relation(R) is identified between "lumina" (light) and "slabă" (dim), where the opinion word is "slabă" and the entity is the noun "lumina".

$$R4: R = \{c.c.m.\}, POS(w_1)=\{adv., vb.\}, POS(w_2)= \{vb., N.\} \quad (2.2.22)$$

E.g. *Gulerul tare și foarte înalt și șapca umflată îi dădeau un aer bărbătesc și elegant. (The very high stiff collar and puffy cup gave him a manly and elegant air.)*

Here the relationship between "gulerul" (collar) and "tare" (stiff) is represented by c.c.m relationship(R), the sentiment word is "tare" and the entity is "gulerul".

$$R5: R = \{n.pred.\}, POS(w_1)=\{adj.\}, POS(w_2)= \{vb.\} \quad (2.2.23)$$

$$R6: R = \{sbj.\}, POS(w_1)=\{N.\}, POS(w_2)= \{vb.\} \quad (2.2.24)$$

The last two rules are used together to determine correctly the sentiment word and the entity. An example regarding this case is *Filmul este frumos. (The movie is beautiful.)*
Here the n.pred. relation is found between the words "frumos" and "este" (is) and the sbj. relationship is between the word "filmul" (movie) and the word "este". In this case the sentiment word is "frumos" and the entity is "filmul".

$$R7: R = \{neg.\}, POS(w_1)=\{neg.\ part.\}, POS(w_2)= \{adj.\} \quad (2.2.25)$$

This relation helps us to find simple negated words like "nu_frumoasa" (not_beautiful). In this case we only considered the simple negations, the ones which contain the word "nu" (not).
Once the opinion words and targets are identified, the next task is to identify the sentiment polarity. In this respect we employ a method relying on SentiWordNet and one relying on search results provided by a search engine.
The first method involves the translation of the lemmatized version of the word in English and assigns a polarity score to the translated word based on the retrieved SentiWordNet scores. The polarity score is computed as the average of all the polarity values found for the word.
The second method involves performing multiple queries on the Bing search engine. The queries have one of the following formats:

- "seed_word și(and) sentiment word" ȘI(AND) entity
- "sentiment word și(and) seed_word" ȘI(AND) entity
- "seed_word sentiment word" ȘI(AND) entity
- "sentiment word seed_word" ȘI(AND) entity

We used as seed words seven positive and seven negative words that proved to be relevant for the English approach. The proposed set of positive seed words is: EXCELENT (excellent), BUN (good), FRUMOS (beautiful), POZITIV (positive), SUPERIOR (superior), CAPTIVANT (exciting), PLĂCUT (pleasant) and the list of negative seed words consists of the following opinion words: ÎNGROZITOR (awful), RĂU (bad), URÂT (ugly), NEGATIV (negative), INFERIOR (inferior), PLICTISITOR (boring), DEZGUSTĂTOR (disgusting).
The polarity of the sentiment is computed as:

$$score = \ln \frac{avgScoreNearPositive}{avgScoreNearNegative} \quad (2.2.26)$$

The average scores represent the average number of co-occurrences of the considered sentiment word with words in the positive/negative seed set.
We experimented another unsupervised approach for opinion holder and target extraction. In this case we propose a solution based on identifying the dependencies between pairs of words and sentences and a list of reporting verbs. The main processes involved in the extraction of opinion entities are text pre-processing (the extraction of POS tag and lemma), reporting verbs and opinion bearing sentences extraction, dependency structure and post-processing modules. For the pre-processing steps we used tools adapted for the Romanian language

(Cristea, 2011) and we also built a list of reporting verbs by translating the English list at (Reporting verbs) and expanding it using a synonyms dictionary. The opinion bearing sentences are selected from the input document based on the number of reporting verbs. The next processing step aims to obtain the dependency structures of the opinion bearing texts. We apply then on the dependency structures Romanian grammar rules and techniques in order to identify the opinion holder and target. The initial set of rules that we apply to achieve our purpose are basic Romanian language grammar rules such as: in a sentence, the opinion holder usually has the syntactic function of subject, and the opinion target usually has the syntactic function of direct object. In order to improve the efficiency of this step, we propose an enhanced set of rules.

2.2.4.2 Document polarity classification

For the document classification task we followed basically the same processing flow as for English documents. However, at each step, some language specific adjustments were necessary. In the feature extraction step we employed a Romanian dependency parser (Cristea, 2011). In the feature selection step we employed a list of noisy words that included Romanian specific words lacking significance from the sentiment analysis perspective.

As for the classifiers we used Random Forest, KNN, Naïve Bayes and SVM in a 10-fold cross validation loop. Based on the preliminary results we only continued with Naive Bayes and SVM, as they proved to be the most suitable considering weighted precision and recall.

2.2.4.3 Experimental results

The experiments were run on a movie review labelled data set in Romanian language, manually extracted (April-March 2014) from several blogs and sites. This corpus is organized into 500 positive labelled documents and 500 negative labels. Each document contains 350 words on average. The documents were manually annotated by taking into consideration the individual grade (in the range from 1 to 10) assigned by the user. The data set is available at http://www.keg.utcluj.ro/datasets.php.

The document classification results using the two polarity identification methods are depicted in Table 16 and Table 17.

Table 16 Classification results for the SentiWordNet approach

Algorithm	*Weighted precision*	*Weighted recall*
Naïve Bayes	81.8	81.8
SVM	79.5	79.5
Random forest	73.5	73.2
KNN	67.1	64.6

Table 17 Classification results for the search engine approach

Algorithm	*Weighted precision*	*Weighted recall*
Naïve Bayes	73.6	74.4
SVM	72.9	70.5
Random forest	72.3	70.3
KNN	64.5	62.5

We tried to improve the obtained results by performing a feature selection step that provided the following outcome for the two approaches:

Table 18 Classification results with feature selection for the SentiWordNet approach

Proc	No.of features	Precision		Recall		Fβ (β=0.5)	
		NB	SVM	NB	SVM	NB	SVM
SW	2195	68.8	69	77.2	70.6	72.7	69.8
SW+FS	372	80.5	80.6	83.8	77.6	82.1	79.1

Table 19 Classification results with feature selection for the search engine approach

Proc	No.of features	Precision		Recall		Fβ (β=0.5)	
		NB	SVM	NB	SVM	NB	SVM
SE	3177	56.3	63.9	70.2	43.6	62.5	51.8
SE+FS	389	69.1	65.4	88	86.8	77.4	74.6

2.2.5 Conclusions

The field of opinion mining involves several challenges. We proposed so far some solutions for the specific task of opinion extraction from English documents relying on an original set of meta-pattern features that proved to be performant. The purpose of the three meta-feature classes that we propose is to boost domain-independence while increasing the degree of generality. Sentiment lexicons provide a basis for the analysis. They encapsulate a sentiment-friendly vocabulary. Part-of-speech patterns reflect syntactic constructs that are a good indicator of polarity. Finally, polarity histograms provide an insight in the distribution of polarized words within the document. All three interact in order to associate sentiment polarity to a document.

We also tried to transfer the developed models for Romanian documents, by identifying the most relevant differences and adapting the solutions accordingly. The obtained results are promising yet less performant than the ones obtained for the English documents.
Our proposed solution is inspired by the approaches presented for opinion extraction from NL texts written in English. The idea was to adapt the rules identified in (Lu, 2010) for English language, considering that the order of the words in sentences is different in the Romanian language. The SentiWordnet solution that we proposed was adjusted by translating the Romanian opinion words into English using the Microsoft Translator API. Our solution was very little affected by the performance of this tool because we only translated processed data (opinion words) not the whole text in order to avoid poor translation. Compared with the results obtained in (Lu, 2010), our approaches using SentiWordnet and Bing search engine, have similar results. When applied the pruning process we obtained a precision score lower by only ~5% then the results in (Lu, 2010).
For future development, at the document level, we intend to extract bigrams to increase the precision of extraction in case of multiple words that together express a single opinion.

2.3 Unsupervised opinion mining approaches

2.3.1. INTRODUCTION

Since supervised approaches are relying on training classifiers on opinion documents, the resulted classifiers are rather domain sensitive meaning that they do not perform equally well when classifying opinion documents from a completely different domain. Some of the reasons are the different vocabularies used in different domains or, even if the same

vocabularies are used, some opinion words might have different polarities in different domains. Therefore, a challenging research area is concerned with domain adaptation.
The objective of domain adaptation would be to get classifiers to perform similarly in a different domain than the training data domain without using training data from the new domain. In this respect one of the first approaches was presented in (Blitzer, Dredze, & Pereira, 2007) and employs a structural correspondence learning (SCL) algorithm based on the selection of pivot features that are used to link the source and target domains. Pivots are selected not only based on their common frequency but also according to their mutual information with the source labels. Second, a measure of domain similarity that correlates well with the potential for adaptation of a classifier from one domain to another is proposed. (Li, 2008) proposed a training approach with data from multiple domains. A multiple classifier system (MCS) framework is presented that integrates a new combining method, called Multi-label Consensus Training (MCT), to combine the base classifiers for selecting "automatically-labelled" samples from unlabelled data in the target domain. (Chen, 2010) presents a semi-supervised classifier RAEMNB that gets trained with a large amount of training data from the source domain and a small amount from the target domain. The target training data is used not only for training the NB classifier but also to adjust the ratio of the predicted positive/negative instances consistent with the distribution of the target domain. A deep learning approach is proposed in (Glorot, 2011) designed to use unlabelled data to extract high-level features from reviews. This is done in an unsupervised fashion from the text reviews of all the available domains using a Stacked Denoising Autoencoder (SDA) with rectifier units for the code layer. In a second step, a linear SVM classifier is trained on the transformed labelled data of the source domain. (Xia R. C., 2013) proposed a comprehensive method, called feature ensemble plus sample selection (SS-FE), for domain adaptation in sentiment classification that includes adaptation at both label and instance levels. The labelling adaptation method based on feature ensemble (FE) relies on the observation that, features with different type of POS tags have a distinct change in distribution in domain adaptation. The sample selection method based on principal component analysis (PCA-SS). One declared shortcoming of the approach is that the training samples are selected in a "hard" way, which is sometimes too arbitrary. A "soft" manner, which assigns a sampling weight to each of the training samples, could be more promising. Reference (Fang, 2014) proposed a hybrid approach that integrates the sentiment information from source-domain labelled data and a set of preselected sentiment words.
However, there are researchers who claim that the idea of "adapting one or multiple domains with plenty of labelled sentiment polarity data to one domain with little labelled data requires new and sophisticated algorithms" should be re-examined (Mansour, 2013). The authors evaluate four domain adaptation techniques on a wide variety of domains in two major groups of state-of-the- art datasets. The results of the experiments show that simple domain adaptation techniques like the all- in-one classifier are not necessarily worse, but sometimes comparably if not better than more sophisticated domain adaptation techniques.
Another aspect we tried to address is the sentiment analysis of social network postings such as tweets. Although there are many research efforts in the field of sentiment analysis, not all the approaches are suitable for social network content. The reasons for that are manifold: (i) one aspect is the fact that the objective of a review is to provide opinions on a target while in social network postings, such as tweets, people might just want to share news/events, to express their emotions like happiness or sadness that are not necessarily oriented towards a certain target (ii) reviews are usually written correctly, in complete sentences that allow for traditional Natural Language Processing (NLP). On the other hand, tweets have a limited size (140 characters) and usually contain slang, abbreviations, special symbols, emoticons etc.

Most of the sentiment analysis approaches are aiming to identify the basic components of an opinion: the target (i.e. the identity that is referred), the opinion holder (who expresses the opinion) and the opinion itself. In the case of social network content or postings the holder is by default the current user but, although the posting involves a sentiment polarity, there is not always a target. The user might feel happy because she had a great time or feels loved etc. In terms of sentiment analysis methods, most of the approaches employ a supervised method that involves the existence of annotated datasets that are used to train classifiers in order to correctly classify other new datasets. These approaches usually yield a performant classification especially if applied in related domains. However, supervised approaches are not that useful to classify tweets since there is no relevant annotated corpus and it is very expensive in terms of effort to create such corpus. Moreover, given the diversity of content existing in tweets, supervised approaches are not able to perform acceptable to classify them.
The next two sections present our approaches to the above mentioned challenges.

2.3.2. DOMAIN INDEPENDENT, DOUBLE-PROPAGATION-BASED APPROACH

The results presented in this section represent a domain independent, semi-supervised approach for performing feature/aspect-based opinion extraction on user generated content (Cosma, 2014), (Suciu D.A., 2014). The starting point of the proposed method is a rule-based, iterative technique, called Double propagation, proposed in (Guang Qiu, 2011). Our research in the opinion mining field concluded that an important problem in opinion summarization is caused by domain specific opinion words. The double propagation approach handles very well this problem through its iterative nature. It performs in parallel feature extraction and opinion set expansion. Unfortunately, being a propagation algorithm, it also introduces much noise which has to be removed. Therefore, we use a set of pruning and filtering methods in order to improve the performance of this algorithm, methods which will be detailed in a later chapter.
Our main objective was to create an efficient and reliable system which can perform well on cross-domain corpora thus assuring a semi-supervised approach to the problem of opinion mining.

2.3.2.1 The rule-based double propagation strategy

The processing flow of our approach starts with the common pre-processing steps performed at sentence level: tokenization, lemmatization, part-of-speech tagging and syntactic parsing. First, each review document is segmented into sentences, which are used for discovering words in the tokenizing step. Lemmatization reduces the word to its base (root) form. The parsing step generates syntactic trees for each sentence, given the output of the previous steps. This syntactic decomposition will be used as input for the second main task of the system, the identification of <feature, opinion> pairs.
As the goal is the correct identification of as many opinion words as possible, in order to assign a polarity value to the whole document, the strategy relies on the double propagation mechanism. The main idea of the double propagation algorithm is to boost the recognition rate in one side (opinion words) by identifying many words in the other side (targets) – back and forth. The extraction method is applied iteratively: the found adjectives and nouns are added to the input set, then new features and opinion words are extracted using the existing ones.
Based on the rules used, polarity scores are transferred from targets or opinions to the newly extracted word. The propagation ends when only few or no new entities are identified. In the end, all the positivity scores of the extracted targets are aggregated to form an overall review

score. As more targets identified increase the chance to identify more opinion words, and vice versa, the process propagate information between opinion words and targets using known and extracted (in previous iterations) opinion words and targets until convergence (very little or at all increase in the words set). The key to identifying opinion words and targets is the use of the syntactic relations defined in Table 20. The algorithm can use any number of seed words, for which we aim to identify an optimal number related to the given system configuration. The propagation consists of four subtasks, all based on the rules defined in Table 20:

- Extracting targets using opinion words (rules 1, 2)
- Extracting opinion words using the extracted targets (rules 6, 7)
- Extracting targets using the extracted targets (rule 4)
- Extracting opinion words using both the given and the extracted opinion words (rules 3, 5)

We proposed the set of rules defined in Table 20 starting from the work in (Guang Qiu, 2011) and extending it with additional constructed rules for extracting adjectives as opinion words based on (Turney, 2002) and new original ones for extracting pronouns as targets.

Table 20 Extraction rules

Rule	Used	Obtained	Dependencies
1	OW	T	1.1 $RB \xrightarrow{MR-Rel} NN$ 1.2 $JJ \xrightarrow{MR-Rel} NN$ 1.3 $RB \xrightarrow{MR-Rel} PR$
2	OW	T	2.1 $RB \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} NN$ 2.2 $JJ \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} NN$ 2.3 $RB \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} PR$ 2.4 $JJ \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} PR$
3	OW	OW	3.1 $JJ \xrightarrow{Conj-Rel} JJ$ 3.2 $RB \xrightarrow{Conj-Rel} RB$
4	T	T	4.1 $NN \xrightarrow{Conj-Rel} NN$ 4.2 $PR \xrightarrow{Conj-Rel} NN$ 4.3 $NN \xrightarrow{Conj-Rel} PR$
5	OW	OW	5.1 $JJ \xrightarrow{Conj-Rel} X \xrightarrow{Conj-Rel} JJ$ 5.2 $RB \xrightarrow{Conj-Rel} X \xrightarrow{Conj-Rel} RB$
6	T	OW	6.1 $NN \xrightarrow{MR-Rel} RB$ 6.2 $NN \xrightarrow{MR-Rel} JJ$ 6.3 $PR \xrightarrow{MR-Rel} RB$
7	T	OW	7.1 $NN \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} JJ$ 7.2 $NN \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} RB$ 7.3 $PR \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} RB$ 7.4 $PR \xrightarrow{MR-Rel} X \xrightarrow{MR-Rel} JJ$

We are using the following notations:
OW = Opinion Word ***T*** = Target
In the dependencies column, we denote the POS tags of the words extracted as
NN = Noun, ***JJ*** = Adjective, ***RB*** = Adverb, ***PR*** = Pronoun, ***X*** = Any POS except if it is the same as the one which follows it. Also we denote the relations between words with:
MR-Rel = A relation between an opinion word and a target
Conj-Rel = A conjunction relation between two targets or two opinions words (usually and, or etc.)
In order to cover the cases when targets are inferred using pronouns we introduced rules 1.3, 2.3, 2.4, 4.2, 4.3, 6.3, 7.3 and 7.4.
Rules 1.1 and 2.1 allow the identification of unknown targets, which are nouns, from known opinion words, which are adverbs. Vice-versa, using rules 6.1 and 7.2 we discover unknown opinion words (adverbs) from known targets (nouns).
For example, in the sentence: *The game loads slowly* if *slowly* (adverb) is known, using rule 1.1 we discover *game*. Vice-versa, if we know *game* we discover *slowly* using rule 6.1. The same reasoning applies for rules 2.1 and 7.2, except the opinion word and the target are separated by another word.
Rules 1.3, 2.3 allow the identification of new targets which are pronouns. Vice-versa, rules 6.3 and 7.3 aid in discovering unknown opinion words (adverbs) from known targets (pronouns).
For example, if in the sentence: *It is working wonderfully* we know the opinion word *wonderfully* we can extract the target *It* using rule 1.3 and if we know the target *It* we can extract the opinion word *wonderfully* using rule 6.3. The same applies for rules 2.3 and 7.3, except for the fact that there is another word that connects the opinion word and the target.
Rules 3.2 and 5.2 allow identification of unknown opinion words, which are adverbs, using known opinion words which are adverbs. In the sentence: *The game works great and smooth*, knowing *great* is an opinion word we can discover *smooth* as well.
Rules 4.2 and 4.3 allow identification of unknown targets from known targets. In case of rule 4.2, the known target is a pronoun and the unknown target is a noun, while rule 4.3 is applied vice-versa.
Considering that an opinion word can have either one target or multiple targets joined by a conjunction we added an additional condition for the X part of speech for rules 2, 5 and 7 from Table I, which we refer to as "X" filter. The "X" filter consists not extracting targets using these rules, if X`s POS is the same the target`s POS, because in this case, we would also extract a target using the rules 1, 3 or 6.
As an example, for JJ->NN->NN, rule 2.1 extracts an invalid target, as the actual target is extracted using rule 1.2 using JJ->NN. From experiments, we have seen an improvement in the results, which justify the use of this condition.
For example, in the sentence: *Me and my wife are pleased with the product* if we know that *Me* is a target we can also discover *wife* as a target using rule 4.2 and vice-versa if we know *wife* is a target we can also discover *Me* as a target. The double propagation algorithm is presented in the following pseudo code:

Double-Propagation Algorithm

```
Input: Seed Word Dictionary {S}, Syntactic Trees {T}
Output: All Features {F}, All Opinion Words {O}
Constant: Objectivity Threshold {Th}
Function:
1. {O} = {S}
```

```
2. {F1} = Ø, {O1} = Ø
3. For each tree in T:
4.          if( Extracted features not in {F})
5.                Extract features {F1} using R1, R2 with {O}
6.          endif
7.         if( Extracted opinion words not in {O} and opinion words objectivity <
   {Th})
8.               Extract opinion words {O1} using R3, R5 with {O}
9.         endif
10. endfor
11. Set {F} = {F} + {F1},  {O} = {O} + {O1}
12. For each tree in T:
13.         if( Extracted features not in {F})
14.               Extract features {F2} using R4 with {F1}
15.         endif
16.        if( Extracted opinion words not in {O} and opinion words objectivity <
   {Th})
17.              Extract opinion words {O2} using R6, R7 with {F1}
18.        endif
19. endfor
20. Set {F1} = {F1} + {F2},  {O1} = {O1} + {O2}
21. Set {F} = {F} + {F2}, {O} = {O} + {O2}
22. Repeat 2 until size({F1}) = 0 and size({O1}) = 0
```

Take for example the following sentences: *The laptop is amazing. The processor is fast and games are amazing and fast, running them on this laptop. The display is also responsive and fast.* Considering only *amazing* as an initial seed word, at the first iteration the algorithm extracts *laptop* and *games* as targets and also extracts *fast* as an additional opinion word. At the second iteration, *processor* and *display* are extracted as targets and *responsive* is extracted as opinion word. The third iteration does not extract any new data thus ending the algorithm.

The process of extracting opinion words and targets from a text using syntactic dependencies introduces noise, so we propose two filters to eliminate the noise both in the opinion words list and the target words list. The opinion words are pruned based on their objectivity. The idea behind the filter is to disregard adjectives or adverbs which are not actual opinion words. An adjective or adverb is considered to be an opinion word if its polarity, taken from a lexical resource, is above (in case of a positive opinion word) or below (negative) a certain threshold. This threshold ensures that objective words are not extracted, thus reducing the chances of noise propagating through the algorithm. The finding was triggered in the initial experimental phase, when we noticed its necessity since many adjectives express a property, not necessarily an opinion (first, other, long, etc.).
The Target pruning module filters out targets based on their occurrence frequency. Because in reviews the product and its features occur more often along opinion words than other nouns, they can be pruned after the extraction algorithm is finished by removing the ones not extracted at least t number of times, where t is a target frequency threshold. The value of t which provides the best precision/recall ration has been determined experimentally.
The third component, Polarity Aggregator, performs the task of assigning polarity values to the extracted opinion words and targets. Moreover it generates a polarity summary by aggregating the individual scores. The Polarity aggregator assigns polarities to seed words using a lexical resource described in the results section. Because a lexical resource usually contains multiple polarities for the same word, depending on the context, the resulting polarity is retrieved as the weighted average of all those polarities. The module uses the list of polarity-charged seed words to assign polarities to the entire text in two steps. In the first step, polarity-charged seed words are matched throughout the text. In the second step, the

previously matched scores are propagated in the entire text. Polarity assignment is accomplished with respect to the following rules:

- Opinion words extracted using targets receive the same score as the target
- Targets extracted using opinion words receive the same score as the opinion word
- Targets extracted using targets receive the same score
- Opinion words extracted using opinion words receive the same score.
- If the same target is discovered using different opinion words, the resulting score is the average of the opinion words.

2.3.2.2 Experimental results

To evaluate our strategy we used the dataset proposed in (Hu, 2004) and adjusted it to our needs by manually annotating the opinion words and targets.

The dataset is composed of 5 subsets of documents, four of which contain multiple reviews targeting a different product, and one represents a fraction of the movie reviews from (Taboada M., 2006). The figures regarding each dataset document are presented in Table 21. The annotated dataset is available on our web site (the Knowledge Engineering Research Group) under the DATASETS link.

In the datasets, opinion words are considered to be either adjectives or adverbs and targets either nouns or pronouns. In the case of pronouns, they are denoted as targets, but for any pronoun the actual target is the product inferred. Pronouns are used to extract inferred product features along with their corresponding opinion words.

The identification of syntactic relations between opinion words and product features was performed by making use of a syntactic parser: Stanford CoreNLP , from which we use the fine-grained POS tags that help identify opinion words and targets. For example, comparative and superlative adjectives are more likely to be opinion words than other kind of adjectives.

More specifically, we employed the following specific classes for each POS tag:

JJ → *JJ, JJR, JJS*
RB → *RBR, RBS*
NN → *NN, NNP, NNPS, NNS*
PR → *PRP*
MR-Rel → *acomp, amod, rcmod, npadvmod, advcl, nsubj, csubj, nsubjpass, csubjpass*
Conj-Rel → *conj, conj_and*

For inferring the polarities of the seed words we used SentiWordNet as it offers both polarity and objectivity for each word, depending on its POS tag and context. To achieve seed words context independency, we compute the weighted average score for each adjective considering all the possible contexts.

Table 21 Dataset details

File	Total Words Number	Opinion Words Number	Targets Number	Sentences Number
Apex	12081	401	358	739
Canon	11543	475	405	597
Coolpix	6501	498	359	346
Nokia	9292	504	277	546
Movie	5456	138	121	248

A brief comparison of the result-wise differences bare implementation of the double propagation algorithm, without the inclusion of the proposed filters, POS tags constraints, or proposed grammatical rules for extracting adverbs as opinion words and pronouns as targets is depicted in Figure 10 as initial results. The final results represent the behaviour of the system with all the proposed grammatical relations.

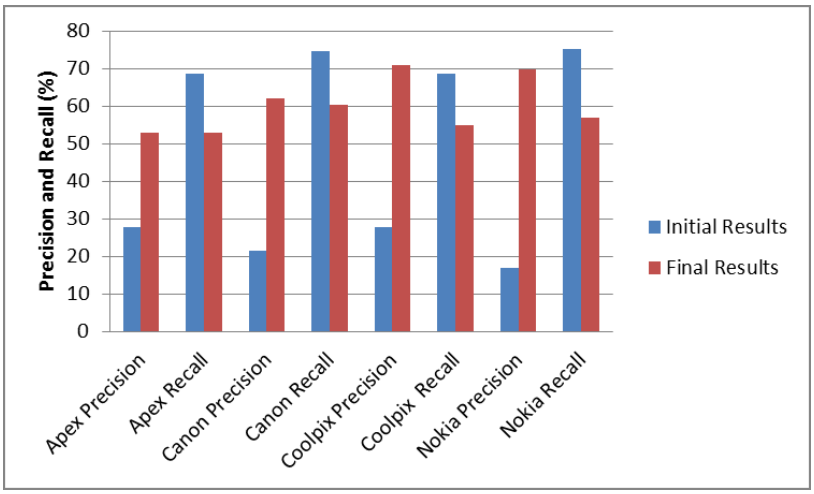


Figure 10 Initial and final results

The next issue to evaluate is the **domain independence.** In this respect we performed tests on reviews targeting different types of products along with the tests conducted on movie reviews, which have a different format and belong to a different domain. The results are presented in Figure 11 and Figure 12, and prove the domain independence of the proposed solution. In Figure 11, the first column from each of the four-set clusters represents the results from tests conducted on product reviews using 6785 seed words. The second column corresponds to tests conducted on movie reviews with the same amount of seed words. The equivalent columns for tests using 2 seed words are the last two of each cluster. Note that the same solution configuration was used for both product and movie reviews (a polarity threshold of 0.01 and a target frequency threshold of 1).

There are generally two types of subjective texts, one which contains only text on topic, like product reviews, and another which is more descriptive in nature, like movie reviews which also describe the plot. In the description, opinions unrelated to the actual target of the subjective text can be conveyed, which affect the extraction process. This behaviour can be seen in Figure 11 on the extraction of movie reviews using 6000+ seed words.

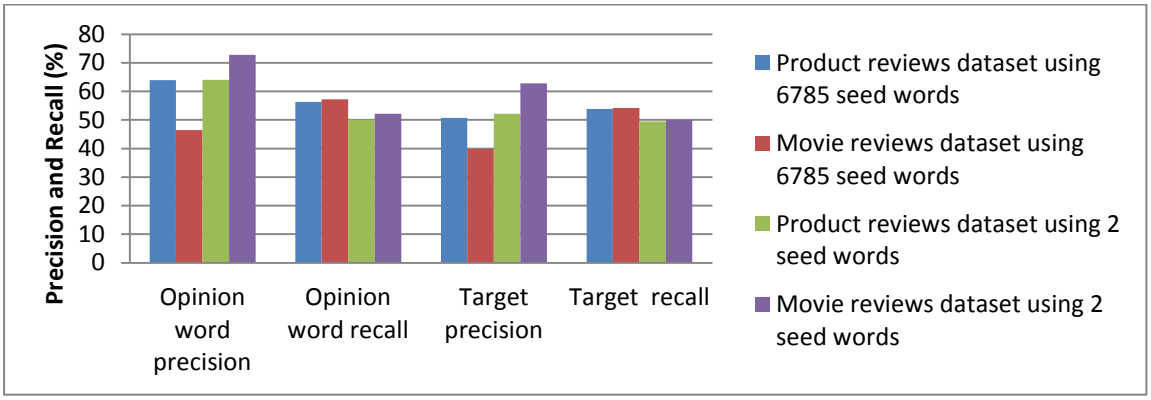


Figure 11 Cross domain evaluation (precision and recall) of opinion words and targets

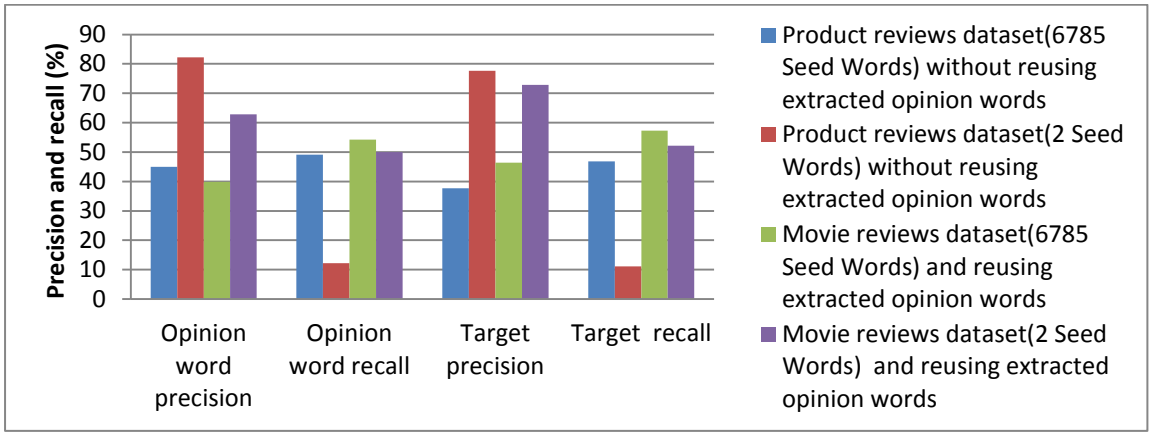


Figure 12 Influence of reusing opinion words as seed words

The usage of only two seed words prevents this unwanted behaviour, as the propagation is generally limited to related targets.

The dimension of the input data also affects the extraction process greatly when two seed words are used, as the propagation process performs poorly on a sparse data set, as can be seen in Figure 12, where the average results "without reuse" depicts the average precision and recall on 8 movie reviews, each of which contain an average of 25 opinion words and 22 targets. As can be seen on the results "with reuse", this issue is solved by reusing extracted opinion words from each text as seed words on all other texts belonging to the same domain, leading to a recall similar as when using a very large set of seed words.

We also performed some experiments in order to **fine tune the parameters.** The results of experimenting with the filtering threshold values showed that for a threshold value of 0.07, the precision increase outweighs the recall drop, but the best results are observed at a threshold value of 0.01. This is due to the fact that increasing the threshold value the number of opinion word omissions increase.

For pruning the targets, we experimented with various values of the occurrence frequency threshold that showed that there is no best ratio of precision vs. recall with the increase in the target frequency threshold value, so the best value can be considered to be 1. Henceforth two best values for the opinion word polarity threshold and target frequency threshold are used, namely 0.01 and 1.

The rules used for extracting pronouns as targets do not have a significant impact the extraction precision for both opinion words and targets, but the increase in recall for both opinion word and target extraction relevant.

A very interesting quest was to find the **optimum number of seed words** and One important finding in our experimental setting is that the number of seed words does not impact the extraction performance significantly, proven by the fact that by using only 2 seed words, i.e. good and bad, results similar to the ones using 6785 seed words were obtained. The small difference in the results presented in Figure 13 proves that no context dependent data is actually needed for a good performance. This behavior is explained by the following two facts: the number of reviews is sufficiently large; there is a high probability that the two – very common – words are used at least once to describe a product or one of its features. After at least one target is extracted, the iterative algorithm finds all the opinion words associated with it. The number of opinion words extracted in this case is close to the one found by using a very large set of seed words. Following this reasoning, we can safely state that this approach is unsupervised.

However, despite the low difference in the results induced by the number of seed words, there is a large difference in the extraction times. The number of seed words dramatically increases

the processing time as can be seen in Figure 14. This is caused by the excessive number checks made by each rule on each possible opinion word. For extracting 400 opinion words using 2 seed words a maximum of 402 comparisons take place, but by using 6785 seed words, over 7000 comparisons are made.

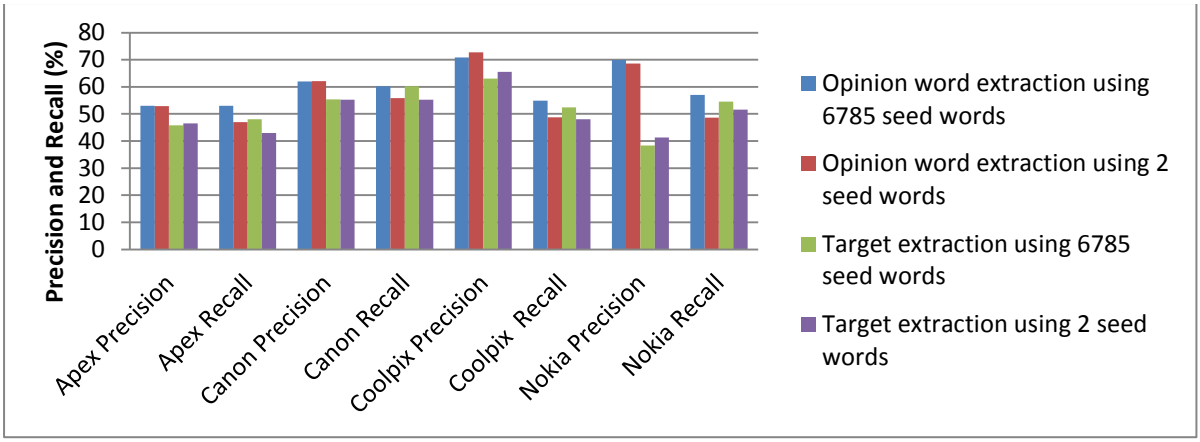


Figure 13 Seed words influence on opinion word and target extraction

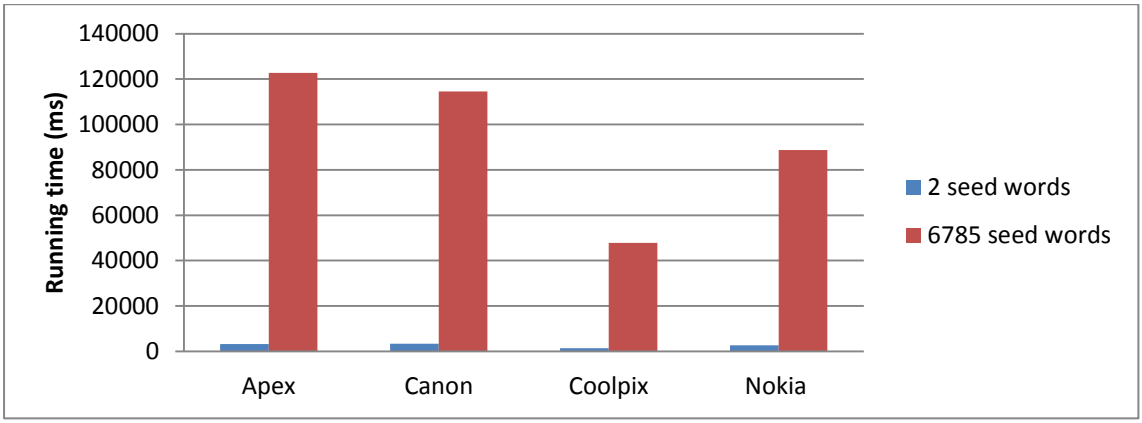


Figure 14 Run times in milliseconds based on seed words

The data set used for evaluating the **polarity assignment** consists of a selection of the first few hundred lines of textual data extracted from the Nikon Coolpix and Canon G3 targeted reviews.

Initial experiments with polarity assignment were performed without taking into account the polarity consistency, that is, without averaging the scores for any of the targets. Precision in that case was just 53%. Applying the consistency rule, which averages the scores for the same target throughout the text, is justified as it improves precision and also evens out the distribution of polarity values.

Another issue that had to be tackled was achieving context independency for SentiWordNet polarity value retrieval. This is due to the fact that SentiWordNet contains multiple entries for the same word, each belonging to a different context and having a different polarity value. To fix this, the total score retrieved for a given word from SentiWordNet is the sum of the weighted averages of its occurrences. The weights decrease with the number of occurrences, as in (2.3.1), as suggested in SentiWordNet.

$$Score = \frac{1}{2} * First\ occurence + \frac{1}{3} * Second\ occurence + \frac{1}{4} * Third\ occurence + \cdots \quad (2.3.1)$$

Further experimentation with the influence of other factors on the polarity assignment module is presented next. There are three factors that influence the precision of scoring: polarity threshold, score threshold and the number of seed words.
Figure 15 depicts the influence of the polarity threshold which has a big impact on polarity assignment precision as it has the power to sooth-out big variations in polarities and filter out inconsistent targets. Using somewhat big values for the polarity threshold, we can obtain 100% precision over non-smooth data sets. The optimal value for this value is determined to be around 0.2 for obtaining high precision values.

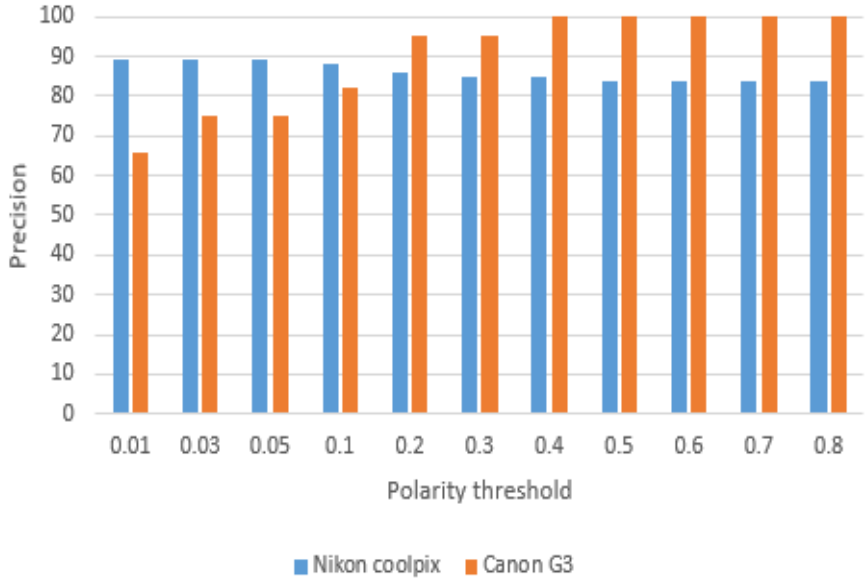


Figure 15 Influence of polarity threshold

In Figure 16 we can see the influence of the score threshold value over the two sets of data. This threshold is necessary since true context independency is very hard to achieve and polarities tend to have variations even in the same context. The polarity threshold was kept to 0.4 because this was the value for which one data set conveyed 100% precision, the target frequency was set to 2 and we used the maximum number of seed words. The optimal value was found to be 0.4. Note that a variation of 0.4 in a scale of 23 entries (-1 to +1) falls very much between most people's subjectivity measures.
Figure 17 depicts the influence of the number of seed words on the polarity assignment precision. This is by far the most interesting result and the most important one as our initial goal was to use just two seed words to obtain comparable results. It was obvious from the beginning that because of the applied rules that ensure polarity consistency throughout the text using just two seed words was not possible since there must be at least one seed word for each major decimal value (0.1, 0.2, etc.) and one for each decimal value in-between and so on. So theoretically, the more seed words the better, but this was not necessarily the case, as Figure 17 shows.

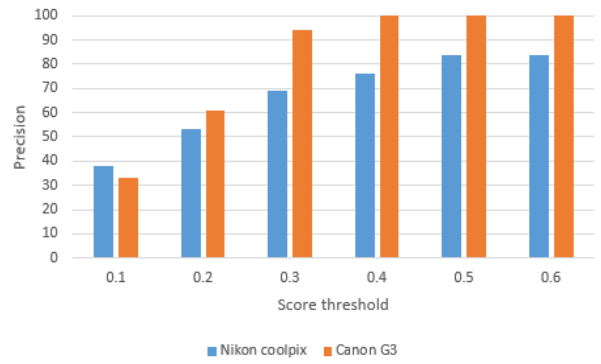


Figure 16 Influence of score threshold

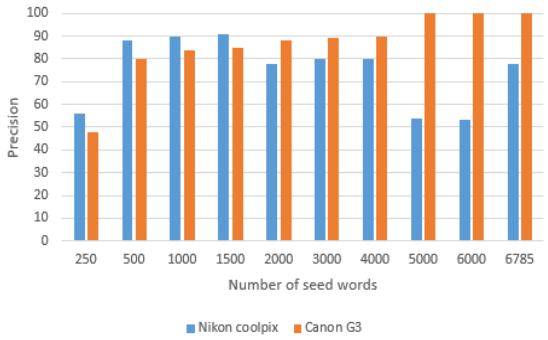


Figure 17 Influence of the number of seed words

Because the Nikon Coolpix dataset contains opinion words conveying mostly the same polarities, using high numbers of seed words introduces noise by "over-averaging" polarities. So naturally, a more specific selection would be beneficial. This is not the case for the Canon G3 dataset which contains diverse opinion words. The best compromise value is at around 1000-1500 words. Notably good results have been obtained using 500 words, out which just 10 were negative words. This is explainable by the fact that angry people tend to use the same negative words over and over again, while happy people tend to use a more elaborate vocabulary.

2.3.2.3 Conclusions

Our work devises a generalized methodology by considering a comprehensive set of grammar rules for better identification of opinion bearing words. We focused on creating a multidimensional configurable system for overcoming the domain-dependency issues which occur in all supervised opinion mining algorithms, by using only 2 class representative seed words. Using thorough experiments we discovered the optimal trade-off between precision and recall, using the opinion polarity and target frequency thresholds. Furthermore, we proved that a larger amount of seed words does not yield a significant increase in recall or precision, making the approach unsupervised and domain independent.

2.3.3. RULE-BASED APPROACH FOR TWITTER

Online reviews are not the only means for people to express themselves, people are now eager to share their experiences and feelings with their friends in real time on social networks such as Facebook or Twitter, too. Although there are many research efforts in the field of sentiment analysis, not all the approaches are suitable for social network content. The reasons for that are manifold: (i) one aspect is the fact that the objective of a review is to provide opinions on a target while in social network postings, such as tweets, people might just want to share news/events, to express their emotions like happiness or sadness that are not necessarily oriented towards a certain target (ii) reviews are usually written correctly, in complete sentences that allow for traditional Natural Language Processing (NLP). On the other hand, tweets have a limited size (140 characters) and usually contain slang, abbreviations, special symbols, emoticons etc.

Most of the sentiment analysis approaches are aiming to identify the basic components of an opinion: the target (i.e. the identity that is referred), the opinion holder (who expresses the opinion) and the opinion itself. In the case of social network content or postings the holder is by default the current user but, although the posting involves a sentiment polarity, there is not always a target. The user might feel happy because she had a great time or feels loved etc. In

terms of sentiment analysis methods, most of the approaches employ a supervised method that involves the existence of annotated datasets that are used to train classifiers in order to correctly classify other new datasets. These approaches usually yield a performant classification especially if applied in related domains. However, supervised approaches are not that useful to classify tweets since there is no relevant annotated corpus and it is very expensive in terms of effort to create such corpus. Moreover, given the diversity of content existing in tweets, supervised approaches are not able to perform acceptable to classify them.
In our work so far we investigated unsupervised approaches for polarity classification of tweets. We also analysed existing lexical, semantic and benchmark data resources that can help to increase the classification performance. We proposed a processing flow and applied our solution to four benchmark datasets in order to compare our results with other state-of-the art solutions.

2.3.3.1. Relevant related work

In (Neviarouskaya, 2011) the authors are concerned with the analysis of message texts in online communication environments in order to identify and interpret sentiments. They propose a novel rule-based linguistic approach namely the Affect Analysis Model (AAM) that was designed to handle informal messages written in an abbreviated or expressive manner, as opposed to correctly written text. The proposed rule-based approach processes each sentence in stages, including symbolic cue processing, detection and transformation of abbreviations, sentence parsing and word/phrase/sentence-level analyses. The sentiments are classified into nine emotion categories (or neutral) based on vectors of emotional words, relations among them, tense of the analysed sentence and availability of first person pronouns. The evaluation of the AAM algorithm showed an averaged accuracy up to 77% on blog posts and 70.2% on fairy tales and news headlines.
The lexicon-based classifier proposed in (Paltoglou, 2012) can be compared with AAM in the sense that is uses emotional lexicons and modifiers but addresses both subjectivity and polarity detection and was tested on three different real-world datasets showing promising results. The features employed are the sentiment baring words as provided by the emotional lexicon having their polarity and intensity modified in case of negation, capitalization, exclamation, emoticon presence, intensification or diminishing. Real-life data was collected from Digg, MySpace and Twitter and SVM and NB were used as classifiers.
Recently, the Semantic Evaluation (SemEval) community has organized a workshop having as topic Sentiment Analysis in Twitter (Wilson T., 2013) where two tasks were proposed: an expression-level task (Contextual Polarity Disambiguation) and a message-level task (Message Polarity Classification). Analysing the results, approaches using supervised strategies such as NRC-Canada (Mohammad S. M., 2013) performed better (i.e. F1-measure of 69% on subtask B) than the only unsupervised method (Ortega R., 2013) that achieved a F1-measure of 50% on subtask B. A novel approach (Bravo-Marquez, 2013) combines aspects such as opinion strength, emotion and polarity indicators, generated by existing sentiment analysis methods and resources, and shows significant improvement in Twitter Sentiment Classification tasks such as polarity and subjectivity identification.
The rest of the participant systems were based on approaches using semi of fully supervised strategies. Finally, this year a rerun of SemEval-2013 task 2 was proposed (Nakov, 2013) as SemEval-2014 Sentiment Analysis Task 98, with new test data from Twitter and another genre. The strongest team overall was again that of NRC-Canada (Mohammad S. M., 2013).

2.3.3.2. Resources Discussion

Available Lexical and semantic resources

In unsupervised approaches, the importance of high-quality resources is critical since no training data is used to obtain high-quality classification models. Research communities have therefore made great efforts to develop new lexical resources for the purpose of sentiment classification, unfortunately most of them for the English language. (Wilson, 2005) annotated a list of English words with positive and negative sentiment categories, thus creating the Opinion Finder lexicon (OPF). The Affective Norms for English Words (ANEW) lexicon was released by (Bradley, 2009) and further enhanced by (Nielsen, 2011), leveraging the AFINN lexicon for Twitter Sentiment Classification. The well-known and intensively used WordNet lexical database introduced by (Miller, 1995) was extended by (Esuli, 2006) through the addition of sentiment scores to synsets, creating SentiWordnet. Furthermore, SentiWordNet 3.0 (SWN3) was introduced by (Baccianella, 2010) as an improvement of the original. Finally, SentiStrength was leveraged by (Thelwall M. B., 2010) for estimating the sentiment strength and further improved by (Thelwall M. B., 2012) as SentiStrength 2 (SS2). Moreover, Mohammad and Turney (Mohammad S. M., 2013) have recently released NRC, a lexicon resource for estimating emotions. NRC contains a number of English words tagged with emotion ratings, according to the emotional wheel taxonomy introduced by Plutchik (Plutchik, 2001)

Besides the lexical resources presented so far, another type of resource that would add semantic level knowledge have been developed. These concept-based resources allow for a semantic analysis of the text using knowledge bases such as web ontologies and semantic networks. Concept-based methods allow for a more subtle detection of subjective information which can be expressed implicitly in a text passage. A publicly available resource of this type that can be used to extract sentiment information from common sense concepts is SenticNet 2.

Other approaches address the generation of resources such as (Neviarouskaya, 2011). This work describes methods to automatically generate and score a new sentiment lexicon, called SentiFul, and how to expand it through direct synonymy and antonymy relations, hyponymy relations, derivation, and compounding with known lexical units. They employ four types of sentiment-relevant affixes (used to derive new words): propagating, reversing, intensifying, and weakening. The approach also considers modifiers, contextual valence shifters, and modal operators, which are integral parts of the SentiFul lexicon for robust sentiment analysis.

Compared Lexical resources

We used in our approach the AFINN, OPF, SS2 and SWN3 resources therefore an analysis of their commonalities and differences is relevant.

SWN3 is much larger than the other resources. However, the SWN3 database includes many words only with objective score (positive score = negative score = 0) that are not that useful when assigning positive or negative polarity to tweets. Due to the fact that we consider also the neutral class, our rule-based classifier takes into account the objective score from SWN3 only if the respective word does not exist in the other lexical resources. The AFINN lexicon has another advantage: it contains some words that are not included in none of the other resources. Among these words are found many acronyms, abbreviations and slang words such as “n00b”, “rofl”, “wtf” and others that are often used in tweets.

We also noticed that there exists some support among different resources i.e. words that have the same (positive or negative) sentiment score in all four lexicons. However, there are other words (e.g. “thanks” and “sympathy”) that can have different sentiment polarities in the four

lexicons SWN3 or simply do not exist (e.g. "sympathy" in SS2). These words need to be analysed in context in order to identify their polarity. Table 22 shows the number of overlapping words in the four lexicons.

Table 22 Overlapping lexicons

	AFINN	OPF	SS2	SWN3
AFINN	2,477	1,246	1,214	1,783
OPFIND	X	6,884	1,487	6,199
SS2	X	X	2,545	2,153
SWN3	X	X	X	147,306

Datasets for testing sentiment classification in tweets

Although other datasets are available, we considered four evaluation datasets for our experiments: Stanford Twitter Sentiment Test Set (STS-Test), STS-Gold, Sanders Twitter Dataset and the SemEval-2013 Dataset (SemEval).

The Stanford Twitter sentiment corpus was introduced by (Go, 2009) and consists of two different sets, training and test. The training set contains 1.6 million tweets automatically labelled as positive or negative based on emotions. For example, a tweet is labelled as positive if it contains :), :-), :), :D, or =) and is labelled as negative if it contains :(, :-(, or : (. Although automatic sentiment annotation of tweets using emoticons is fast, its accuracy is arguable because emoticons might not reflect the actual sentiment of tweets. The test set (STS-Test) is manually annotated and contains 177 negative, 182 positive and 139 neutrals tweets. Although the STS-Test dataset is relatively small, it has been widely used in the literature in different evaluation tasks. For example, (Go, 2009), (Saif, 2013), (Speriosu, 2011) and (Bakliwal, 2012) use it to evaluate their models for polarity classification (positive vs. negative). In addition to polarity classification, (Bravo-Marquez, 2013) use this dataset for evaluating subjectivity classification (neutral vs. polar).

The STS-Gold dataset was introduced in (Saif, 2013) and contains 1,402 negative, 632 positive and 77 neutral tweets. In this dataset, tweets and entities are annotated independently, thus allowing for different sentiment polarities for the tweet and the entities contained within it. This is a great support for performance assessment of entity-based sentiment classification systems.

The Sanders Twitter Dataset consists of 5,512 tweets on four different topics (Apple, Google, Microsoft, and Twitter). The annotation process resulted in 654 negative, 2,503 neutral, 570 positive and 1,786 irrelevant tweets. We did not consider the irrelevant tweets in the evaluation of our classifier. The tweets are oriented mostly on product reviews, leading to domain dependence.

The SemEval dataset was constructed for the Twitter Sentiment Analysis Task 2 (Nakov, 2013) in the Semantic Evaluation of Systems challenge (SemEval-2013). The same data has also been used in the SemEval-2014 Sentiment Analysis Task 9 as development, training and test data. Participants in the SemEval-2013 Task 2 and SemEval-2014 Task 926 used this dataset to evaluate their systems for expression-level subjectivity detection, as well as tweet-level subjectivity detection. The original SemEval dataset consists of 20000 tweets split into training, development and test sets (Owoputi, 2013). We have used the tweet ids provided by (Nakov, 2013) and managed to download 8,696 twe ets with 3,631 negative, 1,405 neutral and 3,660 positive tweets.

2.3.3.3. The tweets classification process

Our approach is based on an unsupervised strategy consisting of three major NLP tasks (i.e. POS-tagging, cleansing/filtering/correction and contextual Word Sense Disambiguation) and a final tweet polarity classification task.

For the POS-tagging operation we have used the last version of a fast and robust Twitter-aware tokenizer and part-of-speech tagger (Owoputi, 2013) for each input tweet. Besides using an extended tag set specialized for tagging tweets, it helps significantly in determining abbreviations, emoticons, hashtags, slang, and incorrect words. From the tests that we have done for our approach, slang words and expressions are tagged either as "!" (Interjection) or "G" (other abbreviations, foreign words, symbols, garbage).

The output obtained from this first phase is represented by <word, POS> pairs that will be submitted to an extensive cleaning and standardization process in the next phase.

To overcome the size limitation of tweets of 140 characters, Twitter users often include characters or symbols with strong semantic such as character repetitions, emoticons or hashtags while also trying to save space by using abbreviations, nonstandard orthography, misspelled words, slang, URLs, just to maximize the impact of the transmitted information, feelings or opinions. Therefore, some text processing is needed in order to eliminate or recover, if possible, noisy or incomplete information.

First, we remove URLs, re-tweets and user mentions. We also eliminate stopwords by using the Natural Language Tool Kit14 Stopwords Corpus. Tokens containing "#" (hashtags), frequently represent an emotion, thought or opinion regarding the tweet's topic, so we remove only the "#" character. We employ a spelling correction algorithm (Norvig) to bring misspells to a grammatical form. Further, we developed a normalization module to delete repeated letters in a word in order to create a correct English word. For example: "speling" is converted into the correct grammatical form "spelling" and "amaaaziing" to "amazing". However, the repeated letters in "amaaaziing" have a boosting effect on the sentiment score so a reference is kept.

Commonly used phrases (e.g. "ain't") are replaced with their grammatical form ("is not") by making use of regular expressions. For this, we created a dictionary with the most commonly used idioms on Twitter, by extracting them from a dataset of approximatively 150,000 tweets, and added an associated sentiment score (e.g. "can't wait": 3). Two additional resources were leveraged to handle emoticons, slang and abbreviations: two emoticons dictionaries[1][2] and a slang and abbreviations dictionary[3]. Each emoticon, slang and abbreviation was manually annotated with a sentiment score. For example, positive emoticons such as ":-)", ":D" are annotated with sentiment scores of 1 and 2, negative emoticons such as ":-(", ":((" are annotated with sentiment scores of -1 and -2, and words such as "thx", "h8" are annotated with sentiment scores of 1 and -2. The range of the sentiment scores are from -5 (negative) to 5 (positive), just as in the AFINN and SS2 approaches.

The resulted <word, POS> pairs are further fed to the Word Sense Disambiguation (WSD) step. The WSD algorithm considers only the pairs that have a valid POS-tag i.e. it is present in the SentiWordNet database: "A", "N", "R", and "V".

[1] http://en.wikipedia.org/wiki/List_of_emoticons

[2] http://cool-smileys.com/text-emoticons

[3] http://www.noslang.com/dictionary

The goal of the WSD process consists in determining the best <word, POS-tag, sense> match for each of the <word, POS-tag> pairs received as input from the previous step. Lemmatization is applied for a better matching percentage. We employed the WordNet SenseRelate[4] implementation for word sense disambiguation. The algorithm uses measures of semantic similarity and relatedness to perform word sense disambiguation and obtain the contextual polarity of all words in tweets. More specific, the algorithm assigns the meaning to a word that is most related to a given set of words, based on semantic similarity measures. We considered the following works on semantic similarity for our experiments: (Jiang, 1997), (Banerjee, 2003) and (Patwardhan, 2003). Since the approach presented in (Jiang, 1997) is limited to studying noun-noun concept pairs, (Patwardhan, 2003) proved to have a higher WSD accuracy for our classification experiments, thus it was integrated in our system.
The required context for each word sense is given by all the other <word, POS-tag> pairs belonging to the same tweet. The triple <WSD_word, WSD_POS, sense> with the highest confidence score is considered a best match, given a <word, POS> pair, if word = WSD_word and POS = WSD_POS. Further, positive and negative sentiment scores are extracted from SentiWordNet, based on <word, POS, sense> matching. The resulted information is represented as the tuple <word, POS, sense, PosScore, NegScore>.
However, the polarity is strongly driven by negation. Therefore, we introduced a negation detection method for every part of a tweet that starts with a negation word (e.g., don't, wouldn't) and ends with one of the punctuation marks: '.', ',', ':', ';', '!', '?' or any combination and repetition between them and added the "_NEG" suffix to each word that follows the negation word. The list of negation words was adapted from Christopher Potts' sentiment tutorial. Finally, the tuple <word, POS, sense, PosScore, NegScore> is updated with the NEG flag by inverting the scores and polarity of sentiment words contained within it.

In (Saif, 2013) eight publicly available and manually annotated evaluation datasets for Twitter sentiment analysis are presented in detail. Tweets in these datasets have been annotated with different sentiment labels including: Negative, Neutral, Positive, Mixed, Other and Irrelevant. Our approach considers only positive, negative and neutral polarities.
In this phase we determine the overall tweet polarity by leveraging a rule-based classifier. The information obtained from the previous processing tasks is completed with the following:

- Sentiment scores and polarities extracted from the lexical resources AFINN, Opinion Finder and SentiStrength 2, for words that are not found in the SentiWordNet database.
- Score boosting values, for sentiment words that follow terms found in the Booster Word List, contain repeated letters, are capitalized or are in a group of two or more successive words having the same polarity.
- Sentiment scores extracted from the dictionaries implemented for emoticons, slang and idioms.
- Ratio between the count of sentiment tokens for each of the positive, negative and neutral polarities and the total count of sentiment tokens found in the tweet.

We consider the following notations:
w – token from tweet having sentiment/emotion score
score(w, p) – positive sentiment score of token w
score(w, n) – negative sentiment score of token w
score(w, o) – objective (neutral) sentiment score of token w

[4] http://senserelate.sourceforge.net

$\boldsymbol{Pos} = \sum_{w \in tweet} score(w,p)$
if (score(w,p) > 0 and score(w,p) > score(w,n))
$\boldsymbol{Neg} = \sum_{w \in tweet} score(w,n)$
if (score(w,n) > 0 and score(w,n) > score(w,p))
$\boldsymbol{Obj} = \sum_{w \in tweet} score(w,o)$
if (score(w,o) > 0 and score(w,p) = 0 and score(w,n) = 0)

$\boldsymbol{PosCnt} = \sum_{w \in tweet} w$
if (score(w,p) > 0 and score(w,p) > score(w,n))
$\boldsymbol{NegCnt} = \sum_{w \in tweet} w$
if (score(w,n) > 0 and score(w,n) > score(w,p))
$\boldsymbol{ObjCnt} = \sum_{w \in tweet} w$
if (score(w,o) > 0 and score(w,p) = 0 and score(w,n) = 0)
$\boldsymbol{TokCnt} = \sum_{w \in tweet} w$
if (score(w,p) > 0 or score(w,n) > 0 or score(w,o) > 0)

$\boldsymbol{PR} = \frac{\mathbf{PosCnt}}{\mathbf{TokCnt}}$ (positive count ratio)
$\boldsymbol{NR} = \frac{\mathbf{NegCnt}}{\mathbf{TokCnt}}$ (negative count ratio)
$\boldsymbol{OR} = \frac{\mathbf{ObjCnt}}{\mathbf{TokCnt}}$ (objective/neutral count ratio)

Based on the formulas presented in Table 23, the classifier determines the dominant sentiment expressed in the tweets i.e. positive, negative or neutral. The tweets that cannot be classified in one of the above mentioned polarity classes will be considered as containing mixed sentiment polarities and included in the neutral class.

Table 23 Classification conditions for each polarity class: positive, negative and neutral

Polarity	Classification condition
positive	$\frac{\mathbf{Pos}}{\mathbf{Neg}} \geq \frac{3}{2}, \frac{\mathbf{Pos}}{\mathbf{Obj}} \geq \frac{3}{2}, \boldsymbol{PR} > \boldsymbol{NR}, \boldsymbol{PR} > \boldsymbol{OR}$
negative	$\frac{\mathbf{Neg}}{\mathbf{Pos}} \geq \frac{3}{2}, \frac{\mathbf{Neg}}{\mathbf{Obj}} \geq \frac{3}{2}, \boldsymbol{NR} > \boldsymbol{PR}, \boldsymbol{NR} > \boldsymbol{OR}$
neutral	$\frac{\mathbf{Obj}}{\mathbf{Pos}} \geq \frac{3}{2}, \frac{\mathbf{Obj}}{\mathbf{Neg}} \geq \frac{3}{2}, \boldsymbol{OR} > \boldsymbol{PR}, \boldsymbol{OR} > \boldsymbol{NR}$

The processing flow is illustrated in the following example starting from the following tweet:

I'm going to watch #TheHobbit #DesolationOfSmaug for the second time! (^_^) Yup this movie is amaaazing!!! http://goo.gl/akx6ai

First, the POS-tagging step is applied and we obtain the following:

I'm (L) going (@) to (P) watch (V) #TheHobbit (#) #DesolationOfSmaug (#) for (P) the (D) second (A) time (N) ! (,) (^_^) (E) Yup (!) this (D) movie (N) is (V) amaaazing (A) !!! (,) http://goo.gl/akx6ai (U)

After applying the cleansing, filtering and correction step, we obtain the following normalized text:

going (V) watch (V) second (A) time (N) movie (N) is (V) amazing (A)

The emoticon (^_^) (E) and the interjection Yup (!) are looked up in the sentiment dictionaries constructed in this phase. A positive polarity is extracted and also the sentiment scores 0.4 and 0.2.
Further, by applying the Word Sense Disambiguation (WSD) step, the best match for each of the <word, POS-tag> pairs is determined. The context is given by all the other <word, POS-tag> pairs from the same tweet. For example, a part of the WSD output for the word "amazing":

amazing#a#2 : 1.10196327385774 : inspiring awe or admiration or wonder
amazing#a#1 : 0.887472599970041 : surprising greatly

The first choice is considered a best match, followed by a search in SentiWordNet for the positive, negative and objective (neutral) sentiment scores:

positive score = 0.875
negative score = 0.125
objective score = 0

Because the positive score is greater than both the negative score and the objective score, it is the only one taken into account for computing the overall Pos score for the tweet.
The overall measures for the example tweet are the following:

Pos = 2.2, Neg = 0, Obj = 0.6
PosCnt = 6, NegCnt = 0, ObjCnt = 3, TokCnt = 9
PR = 0.66, NR = 0, OR = 0.33

Finally, the tweet is classified as having a positive polarity.

2.3.3.4. Experiments and results
We performed the Sentiment Classification task using our rule-based classifier on four datasets: STS-Test, STS-Gold, Sanders and SemEval. For each dataset we selected only the subset of positive, negative and neutral tweets. The performance measures we considered are accuracy, precision, recall and the F-measure for each polarity class. In Table 24 we present the aggregated results for each polarity class.

Table 24 Classification results

	STS-Test	**STS-Gold**	**Sanders**	**SemEval**
Accuracy	81.908	**83.764**	75.261	78.737
Precision –pos	**81.818**	76.144	54.074	80.475
Recall-pos	**91.443**	86.867	86.140	84.907
F-positive	**86.363**	81.152	66.441	82.632

Precision-neg	85.628	**94.098**	57.305	62.056
Recall-neg	80.790	**83.024**	76.758	73.451
F-negative	83.139	**88.215**	65.620	67.275
Precision-neut	77.165	54.251	**93.257**	85.321
Recall-neut	70.503	**78.362**	72.393	74.644
F-neut	73.684	64.114	**81.511**	79.626

For the positive class, the highest measures are achieved on the STS-Test dataset. It is worth noticing that the per-class performance is highly affected by the distribution of positive, negative and neutral tweets in the dataset. Moreover, by comparing the positive class and negative class measures on the SemEval dataset, we can see a clear performance gain in classification towards the first class. Taking into account the evaluation results and the fact that the most balanced datasets are STS-Test and SemEval, we conclude that our approach is better at detecting positive tweets than detecting negative tweets.
For the negative class, the highest measures are achieved on the STS-Gold dataset. However, the number of negative tweets from this dataset is approximately double as compared to the sum between the positive and neutral tweets. Comparing with the evaluation results from the positive class on the STS-Test, STS-Gold and SemEval datasets, it can be observed that the measures for the negative class are clearly influenced by the imbalanced number of tweets in each polarity class.
For the neutral class, the highest measures are achieved on the Sanders and STS-Gold datasets. Again, notice a performance gain for the Sanders dataset that contains much more neutral tweets than positive and negative. On the rest of the evaluation datasets the results are somewhat modest as compared to the positive and negative classes. However, considering the difficulty of correctly classifying tweets having neutral (objective) polarity, we conclude that our results are good, as compared with the other polarity classes.
As was also done in (Liu K. L., 2012) and (Bravo-Marquez, 2013), we focused more on the accuracy and F1 measure than on precision and recall, because both accuracy and the F1 measure are affected by both false positive and false negative classification results.
In Table 25 below we included also the average accuracy, precision, recall and F-measure. The highest accuracy and recall is achieved on the STS-Gold dataset, with 83.764% and 82.751% respectively. For the other two measures, the highest precision and F1-score is achieved on the STS-Test dataset, with 81.537% and 81.062% respectively.

Table 25 Average measures: accuracy, precision, recall and F-measure

	STS-Test	**STS-Gold**	**Sanders**	**SemEval**
Accuracy	81.908	**83.764**	75.261	78.737
Precision	**81.537**	74.831	68.212	75.950
Recall	80.912	**82.751**	78.430	77.667
F1-score	**81.062**	77.827	71.190	76.511

We compare our work with (Bravo-Marquez, 2013) on the STS-Test and Sanders datasets, with (Mohammad S. M., 2013) on the SemEval dataset. Furthermore, we compared our results on all the four evaluation datasets with the results presented by (Saif, 2013).

Firstly, our classifier outperforms the accuracy precision, recall and F1-score of the Baselines used by (Bravo-Marquez, 2013) for the STS-Test dataset by roughly 4%. On the Sanders dataset, we found the results not so conclusive, especially because of the distribution of positive, negative and neutral tweets.
Secondly, our F-score of 76.51% outperformed the F-score of 69.02% reported by (Mohammad S. M., 2013) on the SemEval (2013) dataset. We consider this to be the most important performance indicator of our rule-based classifier.
Finally, we compare our work with (Saif, 2013) which performed only a binary sentiment classification using a MaxEnt classifier, so the neutral class was not considered in the results. The accuracy and the average F-measure reported for our classifier is slightly better on the STS-Test dataset. On the STS-Gold dataset, we obtained a lower accuracy by 2%, but a higher average F-measure by 2%. Again, the results we report on the Sanders dataset are not so good, given the high number of neutral tweets. Last, we conclude that the results on the SemEval dataset are good – only a 4% difference of the evaluation measures.

2.3.4. CONCLUSIONS

We presented in this chapter our work related to unsupervised methods for sentiment classification. The two main solutions are a double-propagation inspired approach that was adapted to rely only on two seed words and a rule-based approach tuned for handling tweets. Both approaches are domain-independent and were tested on benchmark data providing promising results.

3. Knowledge-based solutions

The methods and solutions we developed for the different aspects of knowledge representation, extraction and processing were used for identifying further knowledge in different types of applications. Based on our opinion mining approaches we aimed to extract two types of information: opinion-driven communities (Dinsoreanu M., 2014) and contradictions in opinions expressed by the same holder or different holders (Dînşoreanu M., 2013). Another research direction was related to handling time-series data from the financial domain in order to enhance the forecasting accuracy (V. Ionescu, 2013).

3.1 Opinion-driven Community detection

3.1.1. INTRODUCTION

The main goal of this work is to use the extracted opinions in order to identify opinion-driven communities. The community structure of opinion holders is identified based on the opinion similarity and the social features of holders. A community is therefore defined first by the overall opinion of the community on a target. The overall opinion can be calculated in a straightforward manner as the average of opinion polarities of its members. On the other hand the average opinion value can be confusing since it can be obtained either if most of the members have the same opinion or if half of the members have a strong positive opinion and the other half has a strong negative opinion. Therefore, we define a community by both the average opinion polarity and the variance. The approach relies on the Infomap community detection algorithm (Rosvall M., 2011) and the Multi-Scale Community Detection framework (Martelot E. L., 2013).
In order to apply a community detection algorithm to a data set, we employed a graph representation based on the available opinions and social data.

3.1.2. COMPUTING SIMILARITY

An opinion holder can have opinions referring multiple targets therefore we aim to identify communities based on similar polarity for each target. In this respect we build similarity graphs for each target. A similarity graph is defined by nodes representing holders and weights of the edges between them indicating the similarity of their opinions on the target. Hence the similar their opinions on the same target are, the higher the weight of the edge between the two holders. Since the community structure is determined by the weights of the edges, the way the similarity functions are defined is crucial. In this respect we aimed to experiment and evaluate several similarity functions based on a fundamental metric d (i.e. the Euclidean distance between sentiments), to calculate the weights in the similarity graphs. In the graph building step, we will consider only edges with positive weights. By adjusting the codomain of the similarity function we can determine the edges having the weights <= 0 that are trimmed from the graph. Functions with a codomain which encompasses negative numbers are cutting edges off and have the advantage of producing sparser graphs which are easier to process and in which the community structure should be easier to find. In Table 26 we show the similarity functions that we tested when building the one-dimensional graphs (for one target).

Table 26 Similarity functions at target level

Name	Expression	Exp	Cut-off	Suppression
F0	-d+2	No	No	No
F1	-d+1	No	Yes	No
F2	$F0^{exp}$	Yes	No	Yes
F3	$F1^{exp}$	Yes	Yes	No
F4	$(-d+3)^{exp}$	Yes	No	No
F5	$F4 - 3^{exp} - 1$	Yes	Yes	No
F6	$F4 - 2^{exp} - 1$	Yes	Yes	No

Since d = 0 means equal sentiments and d = 2 means opposed sentiments, we have inverted the codomain so that a higher value indicates greater similarity, as shown in F0. F1 produces values in the range [-1; 1], so only edges with a similarity greater than 0 are included in the graph. When testing functions F0 and F1, we determined that Infomap is less sensitive to the weights and more sensitive to the structure of the edges, so we included exponential versions of these functions: F2 and F3. F2 and F3 produce values in the [0; 2] and [-1; 1] ranges.

As such, for F2, similarities which in F0 were in [0; 1) are made exponentially smaller and suppressed and the rest exponentially larger. The suppressed edges are not removed from the graph but have very small weights compared to the others. Similarly, F3 is the exponential version of F1. F4 is the exponential version of a function with the codomain of [1; 3], to avoid having similarities smaller than 1. F5 and F6 are versions of F4 which perform an aggressive and a less aggressive cut-off, respectively. However, the opinion on one target represents just one similarity dimension. We would like to determine communities of people based on the similarities on several targets. We need therefore an integration strategy of the graphs that were created for each considered target. We analysed four strategies and chose to experiment two of them: Network Integration and Partition Integration (L. Tang, 2010).

Network Integration employs similarity functions that can aggregate data from sentiments on multiple targets in order to develop a single graph based on which com-munities are detected. In our case, the multidimensional data is represented as a sentiment vector for each holder, the elements of the vector being the sentiment of the holder for each considered target. As a

result, the vectors contain elements having values in the interval [-1; 1] enabling a simple approach for calculating the weight for the graph edges by using aggregated similarity functions. We evaluated the aggregated similarity functions presented in Table 27.

Table 27 Aggregated Similarity functions

Table 2. AF0	$\sum$F0	No	No	No
AF1	$AF0^{exp}$	No	Yes	No
AF2	$\sum$F2	Yes	No	Yes
AF3	AF0/C	No	No	No
AF4	($\sum$F1)/C	No	Yes	No
AF5	$AF3^{exp}$	Yes	No	Yes
AF6	$AF4^{exp}$	Yes	Yes	No
AF7	$((\sum F4)/C)^{exp}$	Yes	No	No
AF8	$AF7\text{-}3^{exp}$ -1	Yes	Yes	No
AF9	$AF7\text{-}2^{exp}$ -1	Yes	Yes	No

AF0 and AF2 calculate a sum of similarities based on the previously described functions F0 and F2 for every target the two holders have in common, while AF1 is the exponential version of AF0. AF3 and AF4 calculate the average similarity between the common targets, where C is the number targets the two holders have in common, based on F0 and F1 respectively. AF5 and AF6 are exponential versions of AF3 and AF4. AF7 calculates the exponential average F4 similarities and AF8 and AF9 are versions of AF7 which perform cut-off, similarly to F5 and F6.

In order to include social data, which is also multidimensional, an integration method has to be used to obtain the social community structure. We used the Partition Integration method applied on a graph for each relationship the holders have. Bidirectional relationships define undirected graphs while unidirectional relationships produce directed graphs. Based on the Infomap community detection algorithm we identify one community structure for each target - depending on the opinions. In each of these community structures we can observe communities of opinion holders which have similar opinions on each target. The community detection algorithm is also run on the graph obtained through Network Integration. Here we identify communities of holders with similar opinions on all targets. Similar as for opinions, the community detection algorithm is applied for social data on each obtained graph in the previously described step. However, the resulting community structures are combined using Consensus Clustering as the Partition Integration method. The purpose of Consensus Clustering is to merge multiple partitionings of a set of objects without accessing the features of the objects, as described in (Ghosh., 2003). Since the Normalized Mutual Information (NMI) metric (L. Danon, 2005) for evaluating the common information shared by two partitions is not applicable if we don't know the community structure of the data, we have implemented two consensus clustering methods: Cluster-based Similarity Partitioning Algorithm (CSPA) and Meta-CLustering Algorithm (MCLA). CSPA relies on the number of common partitions as the partitioning criterion while MCLA considers how many objects they have in common. The obtained results are presented in the next section.

3.1.3. EXPERIMENTS AND OBTAINED RESULTS

We employed two quality metrics in order to assess the community detection quality: one for the case when we do not know the actual community structure of the data set (i.e. the network modularity) and one for we know the community structure of the data (i.e. Normalized

Mutual Information (NMI)). Since data with known community structure is generally not available, we developed an algorithm for generating synthetic data sets based on the following parameters: the number of holders, the number of targets, and three real valued parameters in [0; 1]: α, β, and the noise. α represents the maximum allowable similarity between communities (0 means no similarity while 1 means perfect match between community profiles). β controls the belonging degree of holders to the communities and the noise represents the percentage of contrary opinions related to the community opinion.
In order to evaluate the similarity functions shown in Table 26 and Table 27 we have used the opinion generation algorithm. For each generated data set a graph was built on which the community detection algorithm was run. The performance of each similarity function was evaluated by calculating the NMI between the known community structure of the generated data and the identified community structure. This value was averaged on 10 data sets generated for the same parameter combination.
One-Target Similarity Functions: Firstly, we evaluated the functions defined in Table 26. These functions are used when creating a graph from the opinions on a single tar-get. We used the following parameters for generating data sets: 50 holders, 1 target, α = 0.5, β = 0.1 and noise = 0.0, and in each of the three tests the number of holders, alpha respectively beta varied. Figure 18 shows the performance of the functions for the three tests that were run.

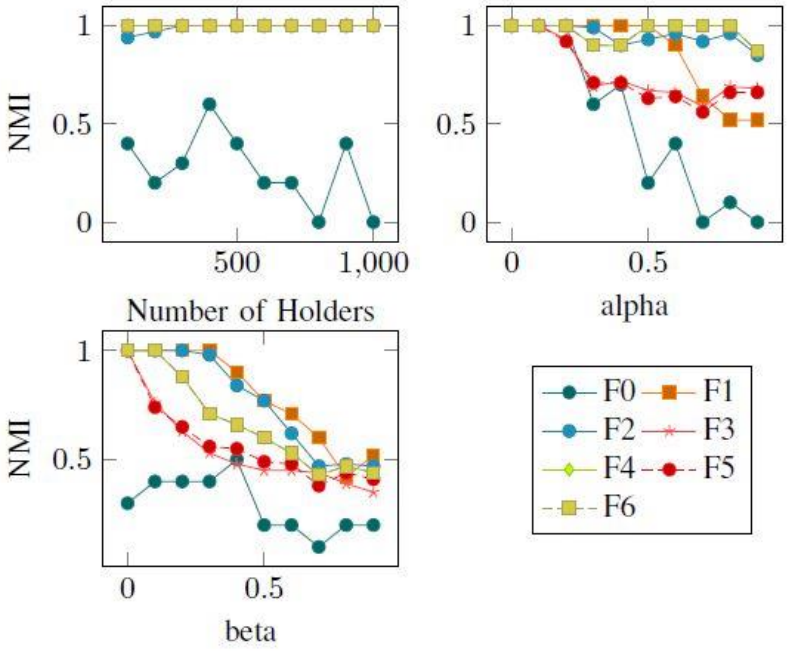


Figure 18 Performance of similarity function for one target

The first test evaluates the performance for a number of holders between 100 and 1000. While F0 has poor and inconsistent performance, the other functions perform very well. The performance of F2 is slightly smaller for a very small number of holders (<200), however for a larger number of holders all functions except F0 manage to perfectly identify the community structure. The second test evaluates the performance of the functions for α varying between 0 and 1. All functions perform very well for a small α i.e. for distinct communities. The behaviour of the performance as α increases is also predictable (i.e. performance decreases) since the more similar the communities are, the more difficult to identify they are. F4 and F6 have very good performance even for a very high α, with F2 being slightly behind. We observe that F4 and F6 have identical performance, which

indicates that the weak cut-off performed by the latter is insignificant. Another interesting finding is that F1, F3 and F5, which perform cut-off have significantly weaker performance than F2 and F4, which do not. Moreover, F3 and F5, which are exponential, have similar performance while F2, which is not exponential, is better performing.
The last experiment evaluates how the performance of the functions varies when β increases from 0 to 1. Again, the performance is very good for most functions for a small β and decreases universally when β increases. F1, which performs cut-off, and F2, which does not, but suppresses edges, perform best. Functions F4 and F6 have identical, weaker, performance again while F3 and F5 behave similarly. We observe that F3, which is the exponential version of F1, performs significantly weaker. This is probably due to the fact that the similarity between members of the same community
decreases exponentially while β increases for F3, which means that nodes are easy to miss-classify.
To summarize our conclusions:

- The number of holders does not affect performance.
- Increases in α and β negatively affect performance.
- Functions that perform cut-off or suppress edges have weaker performance for higher α.
- Functions that perform cut-off have better performance as β increases.
- Functions that both perform cut-off and are exponential have poor performance as both α and β increase.

Multiple-Target Similarity Functions: Our second set of experiments targets the functions in Table 2. These functions are used when building the aggregated graph for Network Integration. We evaluate first the performance of the functions on data sets with 50 holders, α = 0.5, β = 0.1 and no noise, with an increasing number of targets between 1 and 100.

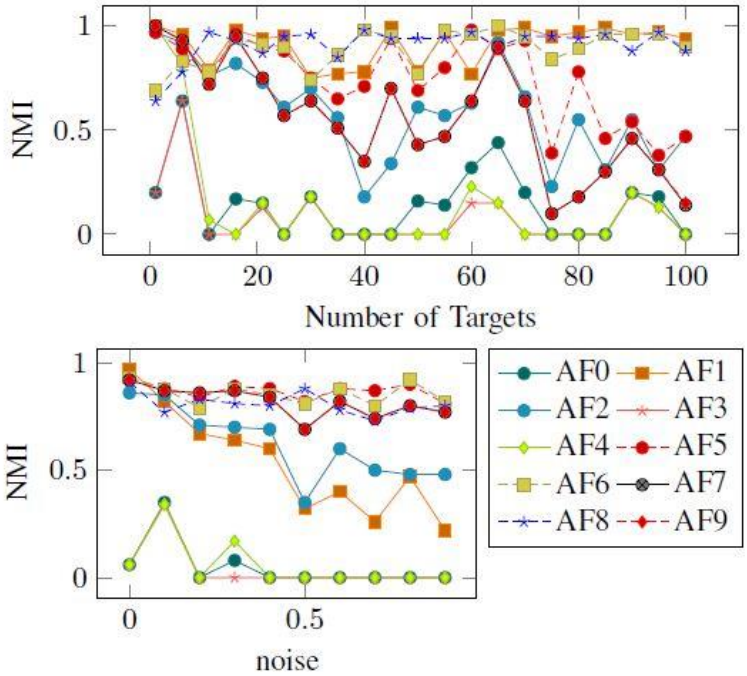


Figure 19 Performance of multiple target similarity functions

According to Figure 19 it can be seen that linear functions AF0, AF3 and AF4 perform poorly for data sets with multiple targets. The exponential functions that do not perform cut-

off (i.e. AF2, AF5, AF7 and AF9) have significantly better performance than the linear functions but their performance decreases with an increase in the number of targets and it is inconsistent for more than 50 targets. The functions that do perform cut-off, (i.e. AF6 and AF8), have very good performance even for a high number of targets. The function AF1 also performs very well, especially for a number of targets greater than 60. This function does not perform cut-off, so we would have expected its performance to be similar to AF2, AF5, AF7 and AF8. It performs however much better than these, the main difference between them being that AF1 does not calculate the average similarity but simply sums the similarities across all targets. The second test evaluates the performance of the functions on data sets with 100 holders, 10 targets, $\alpha = 0.5$, $\beta = 0.1$ and an increasing noise from 0.0 to 1.0. Again we observe that linear functions have very poor performance. Functions which calculate the average similarity (i.e. AF5 - AF9) have very good performance while AF1 and AF2, which simply calculate the sum of similarities, perform significantly worse.

As for the integration strategies in order to aggregate the graphs related to one target, we evaluated the two mentioned strategies, Network Integration and Partition Integration on a generated dataset on 50 holders, with $\alpha = 0.5$, $\beta = 0.1$ and noise = 0.2, the number of targets between 4 and 49. According to the previous results, we used AF6 for the Network Integration approach to build a single graph and run the Infomap algorithm on the graph to obtain the aggregated community structure. For the Partition Integration approach the graphs for each target were generated based on F2 and for the consensus clustering we used both CSPA and MCLA. The obtained community structures were compared with the known ones based on NMI. In Figure 20 the performance of these approaches is shown.

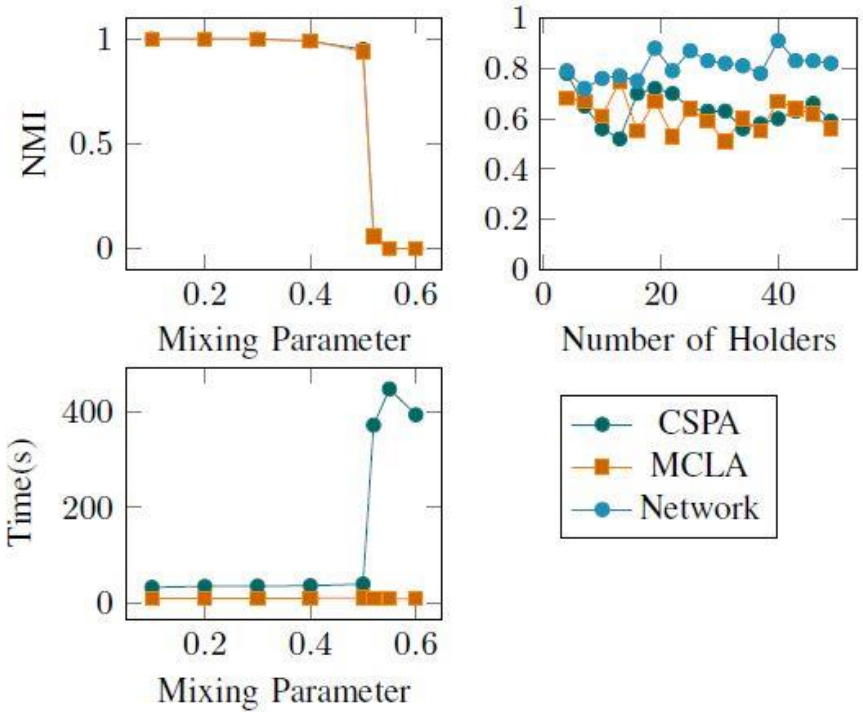


Figure 20 Performance of Integration approaches

3.1.4. CONCLUSIONS

From these tests we can conclude that:

- Linear functions perform poorly for data sets with multiple targets.
- Functions that calculate the average similarity have greater resilience to noise.

Given the results of the tests we use further the function F2 for graphs built considering opinions only from one target, and for Network Integration, we use AF6.
We can conclude that Network Integration performs better than both consensus clustering methods, especially for a high number of targets. We can explain this by the minimal loss of information when aggregating the numeric values of sentiments in the case of Network Integration. Consensus clustering leads to more relevant losses by disregarding the original features that lead to the one-target community structures.

3.2 Opinion-driven Contradiction detection

The contradiction detection problem is another challenge we aimed to address based on the opinion identification and extraction results. We aim for a scalable strategy that allows for accurate real-time tagging of the most frequent types of contradictions (negation, antonym-based contradiction and numeric-based ones) occurring within/across media in large collections, and from various sources. Moreover, our solution is tracking opinion shift over time, detecting individual change of opinion or deviations from the collective opinion. We employed a cloud-based solution to provide the storage infrastructure that allows for scalable opinion analysis. Identification of contradictions within and across documents is a fundamental task in text understanding and falls at the intersection of text categorization, opinion and sentiment analysis, deceptive opinion identification, data mining.
Moreover, when scaling the task and a large volume of data is considered, other research fields (such as parallel/ distributed computing) come to contribute to a successful solution. Contradiction refers to discordant information on an event. To detect contradiction within/across documents, the piece of information has to be decomposed based on the opinion holder. Although not specifically focusing on contradiction detection, (Turney, 2002) can be considered as pioneering work in the field, as semantic orientation can be viewed as part of contradiction identification - opposite opinions as arguments for contradictory information. His method uses mutual information as measure of strength of semantic association between words extracted from reviews. Early work in the field (S. Harabagiu, 2006) has focused on the identification of negation, antonymy and contrast discourse relations as potential sources of contradiction within a document. For the negation part, both directly and indirectly licensed negation is considered in a 4 steps methodology which aims to detect negated events, states and entities. Moreover, a technique to answering negated questions is employed to detect negation.
In the attempt to discover the second type of contradiction, a 5 steps procedure is followed to construct antonymy chains starting from the antonymy relation in WordNet. Finally, discovering contrast relies on the identification of contrast relations based on the semantic and pragmatic signatures (identification of common information two discourse units share plus the information to justify the relation between them) performed within a 7 step procedure. More recent work (M.-C. de Marneffe, 2008) focuses on detecting contradiction across documents. It proposes a definition of contradiction for NLP, argues the need for a more fine-grained distinction of contradictions and outlines the solution which relies on event coreference based on topicality. A typology of contradiction is proposed based on corpora analysis (RTE was annotated for contradiction), with 10 distinct types grouped into 2 primary categories. The proposed solution relies on linguistic analysis, identification of (non-) coreferent events and extraction of features for contradiction detection (grouped into figures-related, structural, antonymy, factivity, modality and relational features). A number of data sources, such as WordNet, VerbOcean are used to enhance the antonymy detection. Several data sets have been annotated and/or enhanced with negative markers. Moreover, a corpus containing a set of 131 real-life contradictions naturally occurring in text was created. With

contradiction defined as conflicting opinions on a specific topic, (M. Tsytsarau, 2011) proposes a scalable solution for sentiment aggregation. Their work considers contradictions in opinions with time as a dimension; thus, asynchronous contradiction (change of opinion) is considered for the first time in the literature. The solution relies on sentiment distribution over documents estimated in terms of topics extracted with LDA applied at sentence level. The polarity is normalized in the [-1, 1] range, and a composite metric to evaluate contradiction, which incorporates the mean value and variance of sentiment distribution, was proposed. Other works in the field report results for very limited scope (narrow domain of applications) and/or only part of the tasks are automatically performed (the rest having to be manually performed).

3.2.1. CONTRADICTION DETECTION APPROACH

Our strategy attempts to identify contradictions by opinion mining across unstructured, free-text documents (i.e. public documents, corporate documents, blogs, social networks). It assumes the identification of the opinion target, the opinion word and other relevant components of an opinion and compares the resulted structure against other opinions stored for the same target. In our approach the opinion retrieved in a new incoming data source is compared against other opinions on the same target expressed in other, previously released data sources. All considered documents are processed and the extracted opinions are stored in an indexed manner, allowing for fast retrieval of opinions of the same holder and/or the same target. For ensuring a fast (real-time) identification of possible contradictions when a new document is entering the system, for all pre-existing documents an expansion mechanism is applied - each disambiguated target and opinion word are extended with context-sensitive synonyms (extracted from external resources). Therefore, when a new document is processed, the extracted opinion is matched against all existing similar opinions for the same and similar holders on the same target. Once a potential inconsistency is detected, it triggers a lexical and semantic analysis of the documents for which the specific conflict was signalled.
We envisioned two usage scenarios: i) an offline scenario that involves the detection of contradictions in opinions given a certain target and possibly holder out of all the stored opinions and ii) an online scenario that provides real-time contradiction detection whenever a new document is available analysing the opinions expressed in the new document against the stored opinions. This involves several steps:
1. Extract the opinion representations of the new document and store them in the repository
2. Extract the relevant stored opinions from the repository, related to the current opinion,
3. Classify the extracted opinions according to their similarity to the current opinion

Our solution is based on a semi-supervised approach that uses machine learning techniques for extracting opinion from documents. We first apply lemmatization and stop words removal, as pre-processing steps, followed by the extraction of features for classification. We have considered several techniques, and build different sets of features, considering unigrams, bigrams (based on word co-occurrence, cosine distance and Pointwise Mutual Information (PMI)). As feature values we have considered both the binary values (to indicate the presence/absence of the feature), and the tf-idf measure of the corresponding feature. We have also considered feature selection techniques, using SentiWordNet as additional resource for filtering features (considering only polarity bearing words in SentiWordNet) and the information gain as relevance measure of the feature. Moreover, as we aim to best identify the opinion bearing words and the entity to which the opinion refers we apply several techniques such as part of speech tagging on each sentence in the text, collecting nouns and noun phrases, and establishing frequent entities.

The information provided by our semi-supervised opinion extraction approach is structured as a tuple for further processing and storage. The tuple has the following format:

$$<t, h, ow, os, di, dp, ts> \quad (3.2.1)$$

where the components have the following meaning:

- *t* is the target (also referred as entity) - a string representing the word on which an opinion is being expressed upon
- *h* is the holder - a string representing the speaker, the person that stated an opinion
- *ow* is the opinion word - an adjective or adverb that describes the entity
- *os* is the opinion score - a real value [-1, 1] that represents the strength of the opinion
- *di* is the doc id - an integer value representing the identifier of the document from which the tuple is extracted
- *dp* is an integer representing the position of the opinion bearing word in the document
- *ts* is the time the opinion was stated

In different opinions the same target can be expressed by synonymic expressions or words. In order to detect opinions that refer the same target, we employ a context sensitive disambiguation of target expressions. Our solution is based on a dictionary and knowledge base approach. In particular, it relies on WordNet hierarchy and the relations defined in this ontology (hypernymy, hyponymy, holonymy, meronymy, entailment, cause, antonymy, attribute, participle-of, pertainymof, also-see, similar-to). We also make the assumption that words that are near to each other in a sentence are more likely to share the same topic. Hence, the relation between the target word and the words in the context is used to determine which of the senses of the target word is closest to most of the context words and that is considered the intended sense. We decided to use a hybrid approach combining three algorithms depending on the analysed parts of speech. We employed the Jiang and Conrath algorithm (Jiang, 1997) for nouns and the Leacock and Chodorow algorithm (Chodorow., 1998) for verbs, because they rely on the taxonomies that can be created in WordNet only using relations specific to these parts of speech. For the other parts of speech we employed an adapted version of the Lesk algorithm (Lesk., 1986.). This is based on the dictionary definitions of the two words to compute the context overlap between them and determine the degree of relatedness of the two words. For the adapted version, we take into consideration not only the definitions, but also other relations defined in WordNet such as hipernymy, hyponymy, meronymy and holonymy.

For storing the opinions and documents, we propose a repository optimized for fast retrieval. This repository integrates an indexing mechanism to increase the performance of retrieval operations. We store the tuples generated by the opinion extraction module and the expansions created by the target/opinion expansion module. For each opinion tuple, the sets T and S are stored in the repository, where T is the list of target expansions for the target of the opinion tuple, while S is the set of synonyms of the sentiment word. The synonyms are stored with their corresponding distances d_l, l = 1;m, where d_l is the distance between the initial target/sentiment word and the corresponding synonym in the context of the initial word.

$T_{jm} = [(t_{j1}; d_1); (t_{j2}; d_2);…; (t_{jm}; d_m)]$

$S_j = [(sw_{j1};d_1); (sw_{j2}; d_2); …; (sw_{jm}; d_m)]$

The data management solution consists of two components: the indexing module and the data storage module, modelled as individual modules. As both modules could be implemented as distributed clusters, there is no one-to-one mapping between any individual nodes of these

clusters. All of this information should be abstracted and these clusters should be able to evolve independently of the other.
In the online processing flow we aim to detect opinions that were previously stated and that are contradicting the current extracted opinions. In this respect, the opinions of a new document are extracted as a set of opinion tuples [o_1; o_2;… ; o_n] which are stored in the repository. The system searches for opinions of the same holder (yet possible described by synonymy). Moreover, opinions are searched also based on their context-sensitive synonyms. The potential inconsistency flag is triggered by the identification of opposite sentiment orientation for the same (or similar) target. Similarity is defined here by synonymy. However, only the opinions with a difference in the sentiment orientation above a threshold are kept as suspected contradictions.
We have considered matching the target of the new input opinion against the target expansions set T_i of the opinions in the repository to improve recall. For each match, if the difference between the sentiment orientation of o_j and the deviation d_i of the sentiment orientation (given by the expansion) of o_i related to target t_i $|so_j - (so_i - d_i)| >$ threshold, implies o_j and o_i are suspected contradictions. The pseudo code for the contradiction detection is described below, where O = [o_1; o_2;…; o_n] represents the set of new opinions mined from the new document, Cand represents the set of candidate opinions extracted from the repository using the holder and the target of each opinion in the new document:

Contradiction Detection Algorithm

```
ContradictionDetection(O, n)
      Contradictions = [];
      s = SentimentOrientation(O)
      for i = 1 to n
            h = HOLDER(O[i]);
            t = TARGET(O[i]);
            Cand = RetrieveCandidates(h,t)
            for j = 1 to size(Cand)
                  sc = SentimentOrientation(Cand[j])
                  if DistanceTest(sc,s) then
                        AddContradiction(Contradictions,O,Cand[j])
      return Contradictions;
```

The matched opinions form the set [ck_1; ck_2; …; ck_n] of possible contradictions. Thus, for each input opinion o_k and set [ck_1; ck_2; …; ck_n] of possible contradictions, a set of pairs of the form [[o_k; ck_1]; [o_k; ck_2]; …; [o_k; ck_n]] is created. This set of pairs can be viewed as a graph, where opinions represent vertices and each pair represents a "suspected contradiction" edge. The set of pairs [[o_k; ck_1]; [o_k; ck_2]; …; [o_k; ck_n]] represents a flag for potential contradictions and is sent for validation in the form [o_k; [ck_1; …; ck_n]], which performs grammar-based analysis to confirm or reject the potential signaled contradictions.
For doing this, a refined analysis is performed on the original documents that contained the given opinions. The phrases containing the opinion words identified in the documents are parsed and the resulted trees are compared in a bottom-up approach considering the first node as the node which contains the sentiment word. Also the holder, entity, attribute, and target nodes are considered as reference nodes if exist. The bottom-up scroll in trees is done in parallel in the two trees representing phrases in the two documents and, at each step, the system compares the two correspondent nodes in trees and determines the relation between them: antonyms, synonyms, hyponyms, holonyms, derived terms, meronyms or hypernyms.

For detecting inconsistency in Negations, the following types of negation terms are considered:
OVERT negation

- overt negative markers: don't, can't, won't, not
- negative quantifiers: no, no one, nothing
- strong negative adverbs: never

INDIRECT negation

- phrasal verbs: deny, fail, refuse, keep from
- prepositions: without, except
- weak quantifiers: few, any, some
- traditional negative polarity items: a red cent, any more

3.2.2. EXPERIMENTS AND RESULTS

The Contradiction Detection performance depends directly on the performance of Opinion extraction. In this section we will not discuss the experiments and results of the opinion extraction tasks since these were discussed in previous sections but we will focus on the specific contradiction detection problem.

3.2.2.1. Experimental setup

Our solution for modelling the repository consists of a distributed NoSQL database and a separate indexing server. We have used HBase for storing all the data in the system. This includes all the opinion tuples, all the documents from which the opinions were extracted and the contradiction graph. To increase the performance we stored the opinions in a separate index. We have chosen Apache Solr for implementing the opinion index. Our sharded implementation allows the distribution of the index over a larger number of machines. A complete list of the tools we chose for the implementation is:
(1) Apache HBase. This distributed datastore provides high performance and scalability when dealing with very large data, billions of rows and millions of columns.
(2) Hadoop File System (HDFS) is required for configuring HBase to run in a distributed fashion.
(3) Apache Zookeeper is an orchestration framework used for coordinating and maintaining the database cluster.
(4) Apache Solr will be used as an external indexing server and will allow complex lookups on the data.
(5) The SpringData project is used for accessing the HBase functionalities in the Java programming language.
The communication with the Repository will be made through a REST API, which will create very little coupling between the repository and modules that will access it.
In order to perform these experiments, we implemented the HBase database using two servers which have been deployed in the cloud. These servers have the following specifications: 10 GB of HDD memory, 2 GB of RAM, a single CPU core, Ubuntu Linux 12.04 OS and a fast network connection, which provided a smooth communication between the servers.
The HBase implementation requires a Hadoop Distributed File System (HDFS) to store the data files on and also an instance of HBase. The HDFS has been configured as follows: the first server has the responsibilities of a master as well as a slave, while the second server had only slave responsibilities.
The HBase has been configured similarly with the first server having master and slave responsibilities while the second one is only a slave. The recommended settings suggest that

the master should have a server of its own and use the whole server's computing power to orchestrate the slaves, but we did not have the necessary hardware available at that moment.
For these experiments the REST API has been deployed on an Apache Tomcat server on a local PC with the following specifications: 4 GB of RAM, two CPU cores, Windows 7 OS. The experiments have been done by using a simple Java console application, which acts as a REST client for the API and calls a set of random queries on the API, which communicates with the HBase cloud servers to return the result.
For evaluating the performance of the indexing mechanism, we have varied the number of partitions for the same index and have measured the performance of system response time for each case.
For the index performance test, each shard was deployed on a machine having 1 CPU core, 2GB of RAM, 10GB SSD for disk storage and Linux 12.04 OS. The orchestration was performed by a dedicated node which ran an instance of Apache Zookeeper.
We have generated the opinions by using a list of 150,000 holders, 2300 targets and 2,500 sentiment words. The sentiment words used were taken from the AFINN list of affectionate words. We generated 100 opinions for each holder. For each of these opinions, a target and a sentiment word were chosen randomly from the word lists. For each opinion, we have also generated 10 targets and 10 sentiment words. We obtained around 5.1 GB of data, consisting of roughly 15,000,000 opinion tuples.

3.2.2.2. HBASE and Solr/Index Experiments
One of the most important queries from the point of view of retrieval time is finding all opinions that match a certain holder and a certain target, because these opinions are the candidates for possible contradictions, as contradicting opinions must have the same holder and must refer to the same target. By running the query to find all opinions by holder and target among 2 million stored opinions, we have obtained an average retrieval time of approximately 2 seconds with a maximum spike of 6.63 seconds as seen in Figure 21

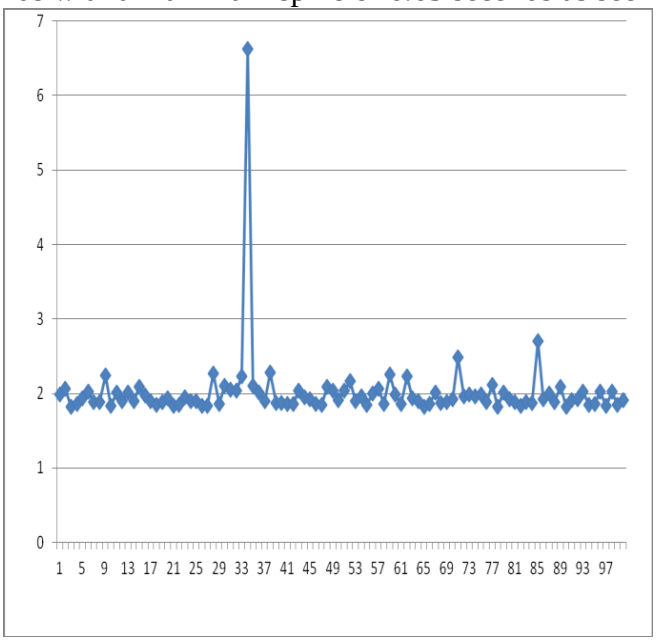


Figure 21 Hbase scatter plot

For evaluating the performance of the indexing mechanism, we partitioned the index in 1, 2 and 3 shards. The times depicted in Figure 22 represent the average over 1000 executions of the contradiction detection flow for a random input opinion. The blue column represents the

average response time for a contradiction detection query when the index resides on a single machine and has a value of 261ms. The red and orange column represent the average response time for an index partitioned into 2, respectively 3 shards. The response time for 2 shards is 192 ms, while the response time for 3 shards is 156 ms. We observe that the response time decreases by increasing the number of shards that host the index. However, it is of note that the improvement decreases with each new shard added.

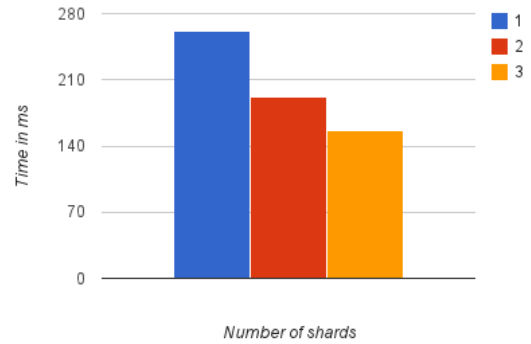


Figure 22 Evolution of index response time given the increase in indexing nodes

We have tested the impact of the index cache to the overall response time of the system. For each query that receives a response, a certain number of the files returned will be stored in the index cache. For smaller indexes, all the response documents could be stored in the cache. However, in the case of larger collections, where the response can have hundreds or even thousand results, it is possible that storing all returned documents in the cache will add too much locality information in the cache. We have tested for 2 different values of the maximum number of documents from the query result stored in the cache per query. Figure 23 presents a comparison between the average response times for 3 different queries for the 2 different values of the maximum number of documents cached. The index contained approximately 4,500,000 opinions generated from 150,000 holders and 30 opinions for holder. The queries we have used to measure the performance are the most common ones for opinion search or contradiction detection.

These queries are:

(1) Retrieving all opinions which have the same holder. This query is represented by the leftmost two columns in Figure 23.

(2) Retrieve all opinion by holder and target. This query retrieves all opinion with a holder h and a target t and is represented by the two middle columns in Figure 23.

(3) Retrieve all opinions by holder, target and target expansions. This query retrieves all opinions with a holder h and a target t or which have t as a target expansions. This query is represented by the rightmost two columns line in Figure 23.

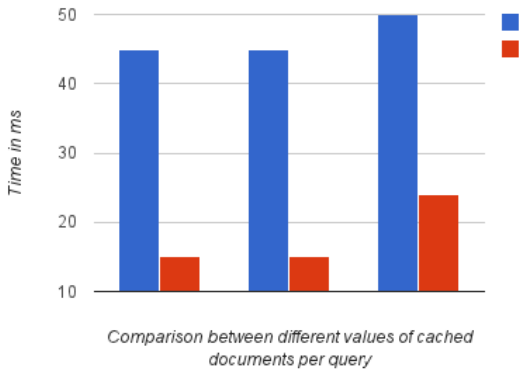


Figure 23 Comparison between different values of cached documents per query

For the experiment we have used a single Solr Node and the SolrMeter benchmarking tool to create 60 queries per minute, each with variables chosen at random. We have performed the experiment on an index where the maximum number of results stored in the cache for each query was 200 and then performed the experiment again on an index where the value for the maximum number of results stored in cache is 2000. The blue colour column represents the cases with 200 maximum results stored in cache for a query, while the red column represents the cases where the maximum number of documents cached was 2000. From these experiments we learn that larger cache value offers better performance for the described dataset.

3.3 Financial market mining

Financial markets have always been one of the typical application areas for a multitude of data mining techniques. This is probably due to the high availability of large amounts of data and the temptation of huge financial gains. Despite these facts, the decades of research have not led to any real breakthrough, i.e. a data-mining based trading strategy that will consistently identify profitable trades.

Some of the most relevant challenges in financial data mining are related to the assumption that financial time series can be explored in order to develop models for financial forecasting. Different types of neural networks were used in order to predict future price or price direction based on the assumption that similar input time series lead to similar outputs. In reality, due to the non-linear, non-stationary, noisy character of the financial market time series it is very difficult for any of these approaches to consistently produce high accuracy market forecasts.

Attempts have also been made in order to identify correlations among different markets, i.e. different time series, in order to determine if an event occurring in one time series can be the cause for an event occurring in another one. In addition to these aspects, another issue of the regular mining approaches is that they provide little means for the trader to provide his own input to the time series analysis/prediction process, and mostly function as autonomous "black boxes", which makes it very difficult to understand the reasoning behind their decision making process.

However, by leveraging focused domain knowledge with smart mining techniques, important gains can be obtained that ultimately result in an improved trading experience.

3.3.1. OUR APPROACH

Much of the difficulty of the prediction task at hand is due to the inherent complexity of the time series forecasting problem. Furthermore the dynamic character of financial time series

makes it nearly impossible for a predictor to consistently perform accurate predictions. Thus, in order to overcome these issues, we propose a different approach for market state classification that involves:

1) Timeseries Sampling

Instead of using the raw price time series as input for the predictor, a sampling of the time series is performed. For each sample only specific features of the market are extracted. Every sample would thus contain the information that defines the "state" of the market at that specific moment. In the obtained set of samples, there no longer exists any temporal interdependency among the individual samples, since each one of them contains all the information that fully defines the state of the market at a certain time.

2) Feature Selection and Evaluation

The actual evaluation of a certain feature within a sample, is performed by analysing the past market evolution over the last periods. The number of periods that are being analysed is a variable of the classification problem that will be referred to as the analysis "window".

The choice of features contained within a sample, representing the market "state", depends on the trading strategy that is being modelled. This approach allows for the concept to be applied to various trading strategies and for the trader to provide his own input to the prediction problem by deciding what features to use.

3) Sample Evaluation and Annotation

The samples which have been obtained from historical data can be evaluated by analysing the subsequent evolution of the market. As a result of the evaluation the samples can then be classified as either positive or negative. The actual choice of annotation algorithm also depends on the trading strategy. A simple example would be to classify a sample as positive if a trade executed at that moment would have rendered a profit, and as negative otherwise.

The set of annotated samples could then basically be used to train a classification model in order to enable a subsequent evaluation of future market evolution. One can see that, by using this approach, the initial time series prediction task has been transformed into a (simpler) problem of sample classification.

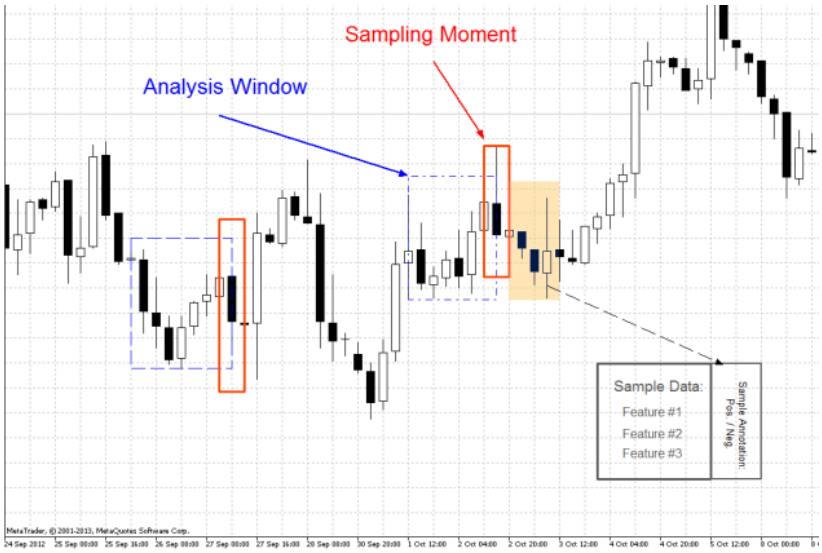


Figure 24 Sample annotation through analysis of the subsequent market evolution

We detail further our approach:

1) Sliding Window

In order to perform the sampling of the time series, the price data is traversed using an imaginary "sliding window" (the same window mentioned earlier while describing the feature evaluation process). An individual data instance is collected for each window position. The attributes contained within each collected data instance are in fact the technical equivalent of the market features described earlier. They collectively describe the current "state" of the market and are determined quantitatively through analysis of the time series evolution within the current window.

The "sliding windows" might be overlapping or not, and might be of fixed or variable size, depending once again on the strategy that is to implemented, e.g.: There is no need for a short-term trading strategy to consider any aspects of the price time-series going farther back than several periods. The situation might be completely different with strategies for long-term trading, where it is desirable to have an overview of the "big picture" of the market. In this situation it might be advantageous to have variable-size sliding windows which can extend over the entire span of the last market trend.

2) Attribute extraction

Any aspect of the market which is considered to be relevant for the trading strategy that is being implemented has to be reflected in some form in the attributes that make up each data instance. The choice of features and the method through which these are quantified and stored in the instance attributes play a decisive role in the success of a certain strategy implementation. This is because the instance's attributes are the only information that the predictor has access to while attempting to classify the instance.

Modelling the various aspects of the market which are relevant to the trading strategy mathematically and representing this information in a form that is usable for the predictor is without doubt the most challenging part of implementing the concept presented here. The task requires a combination of domain and technical knowledge in order to determine the most appropriate technical representation of these market aspects.

A classification resulting in a low accuracy might either mean that the features of the market have been poorly modelled and/or quantified, or it might imply that the current choice of features is indeed fundamentally inappropriate for the currently used annotation algorithm. This is an example of how the state classification approach can be used to infer new domain knowledge from the classification result.

3) Normalization

Also related to the task of proper attribute quantification, another common issue when it comes to mining problems is that of input data normalization. In this case the aim of the normalization step is to eliminate any bias due to differences in time spans and value, in order to allow for a comparison of attributes across different instances.

On one hand typical time series normalization techniques (Min-Max normalization, Z-score normalization) are not applicable to financial time series due to their dynamic, non-stationary character. On the other hand, considering the initial assumption that has been made, that for a certain sample, the market state is only determined by the past evolution of the market from within the analysis window, the appropriate approach is to perform the normalization strictly within each window: The attributes of a certain instance are normalized with respect to the sliding window corresponding to that instance.

As a consequence of using this approach, the bias introduced by differences in time spans and value is cancelled, and attributes of different data instances can be compared to each other successfully.

4) Annotation algorithm

Depending on the trading strategy used, a certain annotation algorithm will have to be implemented in order to classify the test instances for the training phase. The annotation will

typically occur by analysing the subsequent evolution of the market (after the moment at which the data instance was sampled). In order to obtain positive classification accuracy, the criteria by which the annotation is performed should also be inferable by the classifier through analysis of the instance attributes. As already mentioned, a secondary aim of the classification process might be precisely to test this correlation between certain attributes and the chosen annotation criteria.
The overall approach is depicted in Figure 25.

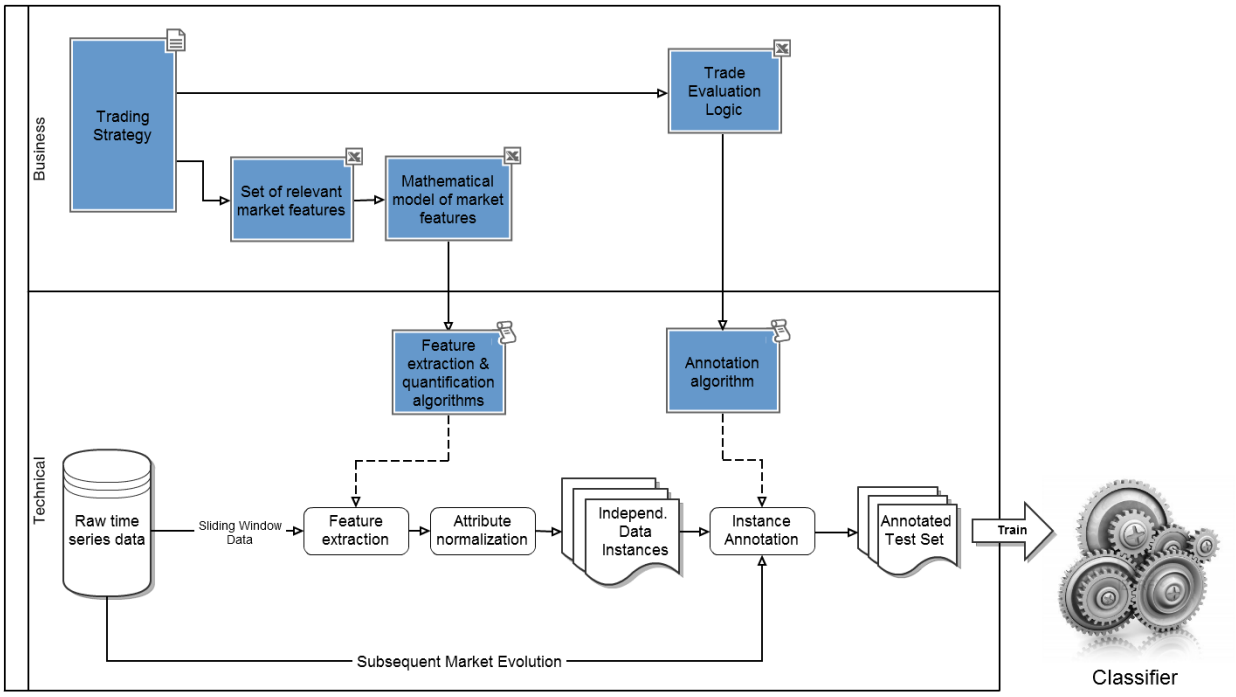


Figure 25 Overview of the activities and artifacts associated with the state classification concept, and their relationship to each other

3.3.2. IMPROVING THE RANGE BREAKOUT STRATEGY

We have performed several studies in order to analyse the benefits that can be obtained by applying the market state classification approach. Results of the studies have been varied, underlining once again the fact that the choice of features and the method through which these are modelled and quantified are key factors that influence the final classification accuracy.
In one of the case studies performed, a commonly used trading strategy has been used in order to demonstrate how applying the concept of market state classification can lead to an increase in trading performance: first, the trading strategy has been simulated on a set of historical market data, recording its results. Afterwards the state classification technique has been applied, in the attempt to obtain improved trading performance. Finally, the results have been compared and conclusions have been drawn.

A. The Range Breakout Strategy

The range breakout is a strategy commonly used when trading financial markets. It is based on the belief that when the market breaks out of a period of consolidation, it is more likely that it will continue its movement in the direction of the breakout. The lower range boundary is called the *support* level, while the upper is the *resistance*.

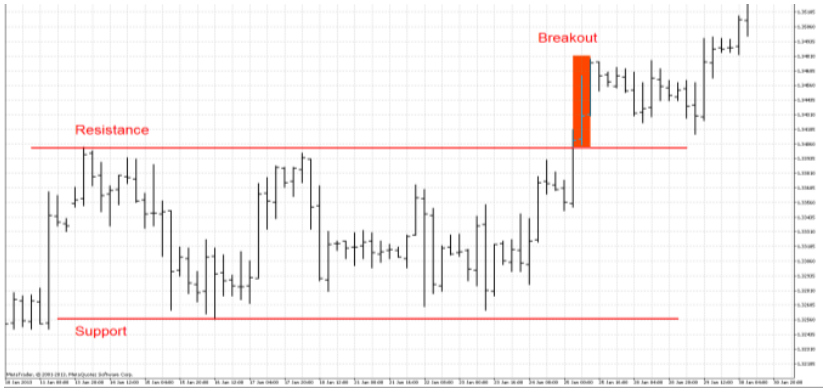


Figure 26 Range Breakout

B. Trade simulation

Based on the Range Breakout theory, a simple trading strategy has been implemented in order to make trade automation possible for simulation purposes. The system would open a trade in the direction of the breakout, every time a breakout occurred. Trailing stops have been used in order to exit the trades; by doing so, the trade is closed automatically as soon as a larger price pullback occurs. A simulation has been performed using historical data of the EURUSD foreign exchange rate from 01.12.2008 to 02.05.2013. At a 4 hour timeframe, this translates to about 2100 periods.

By simulating the range breakout strategy over the specified period, 857 breakouts have been detected. The usage of trailing stops for exiting these breakout trades has resulted in a total of 268 successful trades with a profit of 21218 points2, and 589 unsuccessful trades, totalling a 27951 point loss. In conclusion, applying a range breakout strategy with a trailing stop exit for the period 01.12.2008-02.05.2013 would have rendered a win rate3 of 31.2%, totalling a loss of 6733 points.

Table 28 STATISTICAL FIGURES OF THE RANGE BREAKOUT SIMULATION

	Number of trades	Total win/Total loss (pips)	Average trade (pips)	Win rate	P/L
Win	268	21218	7.8	31.2 %	-6733
Loss	589	27951	4.7		

At this point it is critical to take into consideration the fact that, taken on its own, the win rate is close to irrelevant when evaluating a trading strategy. The strategies applied by professional traders can have win rates ranging from as low as 10-15%, to up to 80-85%. However, what can make the difference between success and failure in trading is smart money and risk management. One has to consider the fact that a trading strategy with an average risk: reward ratio4 of 1:3 can still be profitable even with a win rate as low as 30%.

C. Applying the market classification technique

Further we wish to verify how the state classification concept can be applied to improve the performance of the range breakout strategy.

1) Heuristic assumptions

By studying the domain literature and through analysis of historical data, a set of heuristic assumptions have been made regarding the range breakout strategy. These have not been proven to be true (statistically or by any other means). However, they might aid in increasing

the trading performance to some degree and have thus been chosen to illustrate how, in conjunction with the state classification technique, such information can be used to increase prediction accuracy. The assumptions state that:
a) Trades have a better chance of being successful as the size of the breakout movement increases
b) Trades have a better chance of being successful as the closing price after a breakout is located farther away from the range that the price has broken out of
c) Trades are more likely to be successful if the breakouts occur in the same direction as the higher-level trend
d) Trades are more likely to be successful if the volume traded during the breakouts does not exceed the average by a considerable amount (This is because higher volumes might be an indication that large players have exited the market and a reversal is imminent)

2) Modelling market features
The next step is to determine how these aspects can be modelled technically and quantified in a way that can be useful for the classifier. In order to demonstrate this process, the approach used to model the assumptions *a)* and *b)* will be presented shortly.
For assumption *a),* it is necessary to define a measure for the *movement size* of a period. In this case, the size has been defined as the absolute difference between opening and closing price:

$$M_i = | C_i - O_i | \tag{3.3.1}$$

where:
C_i – closing price at period i
O_i – opening price at period i
M_i – calculated movement size for period i

As described in the theoretical part of the paper, if normalization of a feature is necessary, it will be performed by taking into consideration the size of the sliding window.

$$M_i = (M_i - \mu_{i,window})/\sigma_{i,window} \tag{3.3.2}$$

where:
window – size of the sliding window
$\mu_{i,window}$ – mean movement for interval [i,i+window][5]
$\sigma_{i,window}$ – std. deviation of movement for interval [i,i+window]

In order to model the second assumption, it is first of all necessary to determine where the range boundaries (i.e. the support and resistance levels) are located. This is done by taking minimum and maximum price values for the period [i,i+window]

$$S_i = \min_{i,window} \tag{3.3.3}$$
$$R_i = \max_{i,window} \tag{3.3.4}$$

More complex approaches have also been tested, which evaluate the relevance of multiple potential S/R(support/resistance) levels, but for small window dimensions, these approaches have not resulted in any significant performance increase. Once the S/R levels have been determined, a function specifying the relative market location at a certain point, with respect to the range boundaries has been defined as:

[5] A time series index of i = 0 corresponds to the current period. Older periods have higher index values

$$L_{i\,=}\,(C_i - S_i)/(R_i - S_i) \tag{3.3.5}$$

With a quick look at (3.3.5) one can notice that L_i is a linear function with L_i=0 when $C_i = S_i$, i.e. the market is at the support level and L_i=1, when $C_i = R_i$. Considering the fact that breakouts (and thus also trades) can occur both to the upside (L_i>1) and to the downside of the range (L_i<0), it is advantageous to treat the two situations uniformly throughout the prediction process. Consequently a differentiation among the two situations might be counterproductive during the classification phase, and with that in mind, the function definition has been adjusted to:

$$L_i' = |2 * (C_i - S_i)/(R_i - S_i) -1| \tag{3.3.6}$$

The range of L_i' is [0, +inf), and a value of L_i'>1 is an indication that a breakout has occurred (either above the resistance, or below the support). Normalization of this feature is not necessary since, because of the chosen calculation algorithm, the relative market location is implicitly normalized to the range of the current sliding window. Figure 27 illustrates the S/R levels for a window size of 10 periods. Below the chart, the L_i' function has been plotted. L_i'>1 indicates that a breakout has occurred.

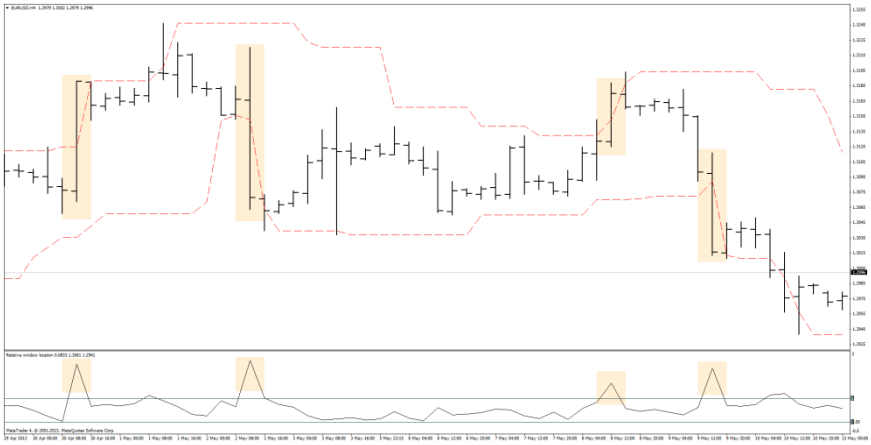


Figure 27 Plot of S/R lines(dotted red lines) and of the L_i' function(below).

3) Instance and Attribute extraction

After having defined all features of the market that are necessary in order to model the assumptions regarding the breakout strategy, these definitions have been implemented in order to allow the corresponding attributes to be extracted from the market. As a result each data instance sampled from the financial time series would have the following attributes:

a) Normalized size of the breakout movement
b) Market position relative to S/R levels (L_i')
c) The previous period's market position relative to S/R levels
d) Volume traded, normalized to the sliding window
e) Correlation of breakout direction with the higher level trend

While performing the sampling of the time series, the results of the breakout strategy have been used as filter, i.e. only those samples have been collected that had been identified as breakouts. As a result 857 instances, each having 5 numerical attributes, have been obtained.

4) Instance Annotation

Since the breakout trade simulation has uncovered which of the breakouts would have been profitable, this information can be used in order to perform an annotation of the test instances. Thus, to each data instance a new binary attribute is appended, representing the class of the instance, i.e. "positive" for profitable trades and "negative" for losing trades.

5) Performing the test

The approach described in the previous sections has been executed using a 10-period sliding window. The obtained annotated test instances have been used in order to train a Naive Bayes Classifier. In order to evaluate the results, 10-fold cross validation has been used. The results of this approach are presented in the next section.

3.3.3. EXPERIMENTS AND RESULTS

It is important to note that there are 2 important aims when applying the state classification concept. On one hand there is the obvious aim, of finding trading strategies with increased forecasting accuracy, which will be discussed shortly.

However, in addition to this aspect, the state classification can also be used to infer new domain knowledge through analysis of the instance attributes and of their correlation to the annotation algorithm. In this case, an Attribute Selection has been performed in order to determine the attribute subset with the highest potential to discriminate among the 2 classes (positive and negative). The subset evaluation has shown the first 2 instance attributes (movement size and relative market location) to be the attributes with the highest discrimination potential.

We will now turn our attention to the actual results of the classification problem.

1) ROC Curve Analysis

The Receiver Operating Characteristic is a metric often used to quantify the ability of a classifier to discriminate among the two classes of a classification problem. Figure 28 illustrates the ROC Curve of the breakout classifier. While an AUC (area under curve) rating slightly above 0.62 indicates that the classifier is far from being ideal, it also demonstrates that there does indeed exist an edge which can be exploited.

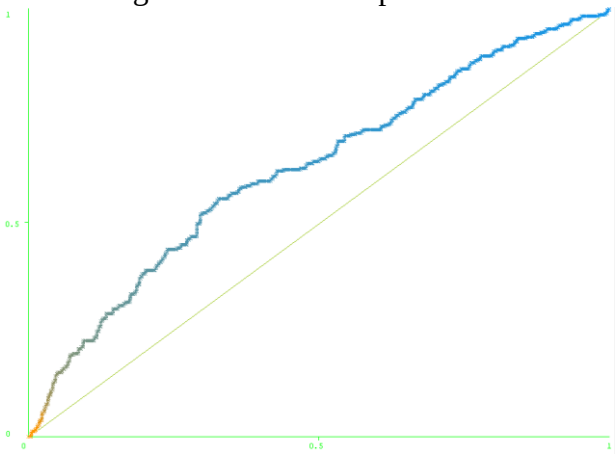


Figure 28 ROC Curve (X – False Positive Rate, Y – True Positive Rate)

2) Classifier Evaluation & Optimization

As always when dealing with a classification problem, a trade-off between the various aspects of the classification result (e.g. precision vs. recall) has to exist. In the next step, the trained classification model has been tweaked with the intent of maximizing various aspects of the predictor and analysing the results.

a) Precision

The maximum precision has been obtained at a threshold of 0.56: 57.5%. In the context of our initial problem, the precision of the classifier would in fact represent the win rate of the trading strategy. A win rate of 57.5 % would have meant that 38 positive trades have been detected (i.e. a recall of only 14.1%).

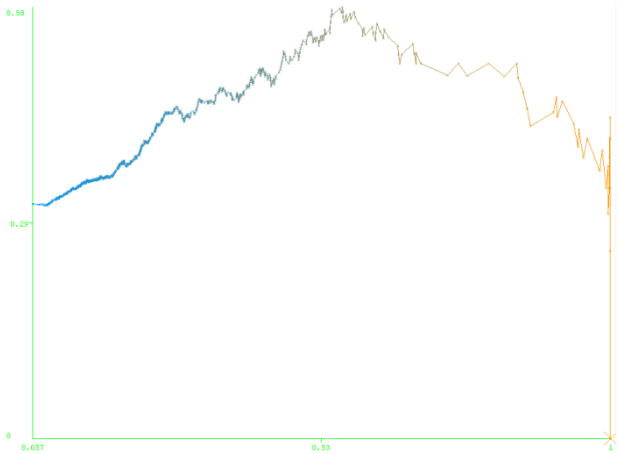


Figure 29 Precision Plot (X – threshold, Y – Precision)

b) F-Measure

A commonly used metric is the F-Measure, which combines precision and recall into a single measure. Maximizing the F-Measure leads to 238 winning trades and 459 losing trades which equates to a 88% recall with a precision of 34%.

c) Cost/Benefit

However, since in fact the aim of any trading strategy is to maximize profit, the classifier optimization should target the same aspect. In classification problems this can be achieved by performing a cost/benefit analysis of the classifier.

After having performed the trade simulation, we have concluded that the simple range breakout strategy used with a trailing stop exit would have produced a 21218 point profit through 268 winning trades, while the 589 losing trades would have cost the trader 27951points. This translates to a statistical average winning trade of 79.1 points. The average loss incurred through a losing trade is of 47.4 points. (As a side note, this corresponds to risk-reward ratio of 1.66)

These facts can now be used as input for the cost/benefit analysis: The benefit of a "true positive" is 79.1 points, while the cost of "false positive" is 47.4.

False positive and false negatives are irrelevant for this optimization scenario since no trade would have been performed in those situations, in which case the equity of the trader would not have been affected in any way.

Figure 30 illustrates the C/B curve, for the previously defined cost matrix. The threshold selected through cost/benefit optimization is 0.29, which implies 139 true positives, 175 false positives, a win rate of 44.2% and a statistical profit of 2699.9 points (well over the 6733 point loss of the initial setup).

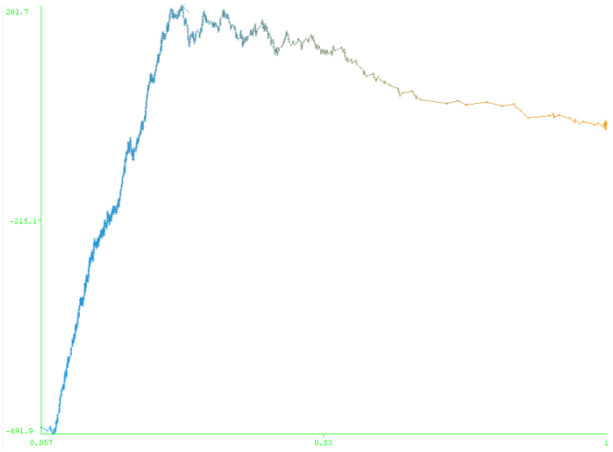


Figure 30 C/B Analysis(X – threshold, Y – total Benefit)

The market state classification technique can basically be applied with 2 purposes in mind:

- Obtaining a high accuracy market forecasting technique
- Inferring new domain knowledge regarding the correlation between various aspects of the market

As the range breakout implementation has demonstrated, if implemented properly, the presented concept does have the potential to provide a substantial increase in trading performance.

3.4 Conclusion

In this section we presented the approaches we developed to use the extracted knowledge in order to analyse it and derive new knowledge. Based on the opinion extraction approaches we designed and developed contradiction detection methods that can be employed in different environments such as: corporate environments to track opinions of the same person/group in time, public/political statements in order to identify contradictory statements issued by the same group, opinion shifts in time etc. Moreover, based on the extracted opinions we can identify communities of opinion holders according to the similarity of their opinions. In the community detection method we integrated also social data besides the opinion information. We experimented several similarity functions with different parameters and identified the best configuration for our datasets.

In terms of financial market state classification, our technique aims to overcome the identified drawbacks of the studied time series based techniques by allowing more flexibility in terms of user-driven input into the mining process, depending on what aspects of the financial market the user considers to be relevant for his own trading strategy.

On the other hand, through this approach the complexity of the mining process has greatly shifted toward the data pre-processing. A particularly challenging task is that of modelling the relevant market aspects from the business domain to the technical space and quantifying them as discriminative instance attributes. As the range breakout implementation has demonstrated, if implemented properly, the presented concept does have the potential to provide a substantial increase in trading performance. Further envisioned developments might include designing automated methods for sampling and feature extraction, more tightly integrated with the activities of attribute selection and instance classification, such that a usage of the state classification concept.

Professional and academic future development plans

As a full time associate professor, the candidate will continue to develop and update the Software Design and Project Management lectures and aims to develop new lectures related to the design of systems handling big data and applying machine learning techniques to big data. The candidate also aims to continue to advise bachelor and masters students in the aforementioned research fields. If the habilitation will be successful, the candidate will be able to advise PhD students. This will lead to a new level of research quality. It also involves an increased effort to attract additional funding in terms of national and/or international research projects. The envisioned outcome of the doctoral research is a set of efficient methods of managing and analysing big data integrating semantic that can be adapted to different domains and applied in real-life systems. The methods will be published in relevant conferences and/or journals.

Scientific future development plans

The future research plans start from the current achievements but also current limitations of the obtained results and aim to overcome them. However, the candidate intends to explore new research directions too, in order to expand the area of expertise and to develop a strong, even if small, research group.

To be more specific, in the field of **knowledge extraction from unstructured data** (text) the main directions are related to:
1) ***unsupervised and domain independent*** approaches for extracting knowledge from text. We explored so far several supervised approaches for identifying and extracting opinions and sentiments in documents such as product reviews, tweets etc. The obtained results were comparable with the ones reported in literature, however, the approaches need a training data set and the developed models are highly-dependent on it. Since in most cases there is no available training data set and the time needed to create one is rather long, the development of unsupervised approaches is of interest. One of the directions we investigated with rather promising results is based on a double-propagation algorithm starting from just two very general seed words for identifying the sentiment polarity of a document. Other directions that can be investigated are related to experimenting with different variations of LDA such as: (i) constrained-LDA that employs constraints in Gibbs sampling to bias the conditional probability for topic assignment or (ii) Multi-Grained LDA to model both global and local topics.

2) ***adapting*** the obtained methods to be applied ***for the Romanian language***. Although good quality results were obtained when extracting knowledge from documents written in English, applying directly the same methodology to Romanian does not yield the same quality in results. Some of the reasons are the lack of high performance tools for obtaining good quality features such as part of speech, the different grammar rules etc. In the context of the current Swara project, our goal is to provide methods for Romanian text phonetic transcription, syllabification, stress assignment, normalization. In this respect we aim to experiment different approaches such as: a look-up approach based on efficient storage and query methods, a rule-based solution and a machine learning solution relying on problem specific methods such as Structured Support Vector Machines. We also aim to develop an exhaustive annotated lexical resource. Another objective in the framework of the Swara project is to develop a text completion algorithm that learns from the user interaction in order to provide

the most probable next word or rest of sentence. We investigated so far different representation approaches such as ternary search tree solutions, signifficance fussytree solutions etc.

Related to the **knowledge extraction from structured data** we aim to focus on:

1) methods for ***storing and analysing Big Data*** while integrating semantic in the analysis process. Traditional storage and analysis mechanisms are not applicable to Big Data, therefore new cloud-based technologies emerged for non-relational storage structures and Big Data analysis solutions. We aim to explore methods for efficient integration and consolidation of heterogeneous data collected from various InternetOfThings (IoT) platforms in order to determine context and analyse the data in a context-meaningful manner.

2) methods for ***storing and analysing data streams*** while integrating semantic in the process. In the context of collecting and analysing large data streams we aim to explore solutions for an efficient real-time analysis of data streams. The main challenges we identified so far are related to the data representation and finding similar subsequences in data (such as range query and k-nearest neighbours) or recurring patterns. The main approaches for time series representation involve methods for dimensionality reduction (such as Fourier transforms, piecewise aggregate or linear approximations or symbolic representations (i.e. SAX format)), segmentation and indexing.
For similarity approaches we aim to investigate data adaptive solutions by employing variable size segments, normalized representations to allow for similarity detection in scaled sequences.
The solutions can be applied in the same IoT domain considering the data continuously collected and transmitted by various sensors and devices. We also envision the application of the solutions in the financial sector for analysing financial time series.

References

Agrawal, R., Gollapudi, S., Halverson, A., & Ieong, S. (2009). Diversifying search results. *Proceedings of the Second ACM International Conference on Web Search and Data Mining* (pp. 5-14). ACM.

Ahmed, A., Low, Y., Aly, M., Josifovski, V., & Smola, A. J. (2011). Scalable distributed inference of dynamic user interests for behavioral targeting. *17th ACM SIGKDD international conference on Knowledge discovery and data minin* (pp. 114-122). San Diego, California, USA: ACM.

Anisiei, A. L. (2014). Extracting opinion holders and targets in Romanian texts. *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on* (pp. 37-42)). IEEE.

Baccianella, A. E. (2010). *SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining.* Valetta, Malta: European Language Resources Association ELRA.

Bakliwal, A. P. (2012). Mining sentiments from tweets. *Proceedings of the WASSA .*

Banerjee, S. &. (2003). Extended gloss overlaps as a measure of semantic relatedness. *In IJCAI (Vol. 3)*, 805-810.

Beneventano, D., & Bergamaschi, S. (2004). The MOMIS methodology for integrating heterogeneous data sources. *Building the Information Society*.

Blei, D. M. (2011, December). Introduction to probabilistic topic models. *Communications of the ACM, 54*(12), 77-78.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003, March 1). Latent dirichlet allocation. *The Journal of Machine Learning Research, 3*, 993-1022.

Blitzer, J., Dredze, M., & Pereira, F. (2007). Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. *Proceedings of the 45th Annual Meeting of ACL*.

Bradley, M. M. (2009). *ffective Norms for English Words (ANEW) Instruction Manual and Affective Ratings.* Center for Research in Psychophysiology University of Florida.

Bravo-Marquez, F. M. (2013). Combining strengths, emotions and polarities for boosting Twitter sentiment analysis. *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining* (p. 2). ACM.

Broder, A. F. (2007). A semantic approach to contextual advertising. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 559-566). ACM.

Broder, A., & Josifovki, V. (2011, Autumn). Introduction to computational advertising (MS&E 239). Stanford, California, USA: Stanford University.

Chakrabarti, D. A. (2008). Contextual advertising by combining relevance with click feedback. *Proceedings of the 17th international conference on World Wide Web* (pp. 417-426). ACM.

Chatterjee, N., & Krishna, M. (2007). Semantic integration of heterogeneous databases on the web. *Computing: Theory and Applications*, 325-329.

Chen, W. Z. (2010). A Text Classifier with Domain Adaptation for Sentiment Classification. In *Information Retrieval Technology* (pp. 61-72). Berlin: Springer Berlin Heidelberg.

Chodorow., C. L. (1998). Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database 49(2)*, 265-283.

Cosma, A. C. (2014). Overcoming the domain barrier in opinion extraction. *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on* (pp. 289-296). IEEE.

Cristea, D. (2011). Romanian Linguistic Resources on Very Large Scale. *Computer Science Journal of Moldova, vol.19, no.2* , 130-145.

Dînşoreanu M., P. R. (2013). A scalable approach for Contradiction Detection driven by Opinion mining. *Proceedings of International Conference on Information Integration and Web-based Applications & Services - IIWAS '13*, (pp. 7–15).

Dinsoreanu M., P. R. (2014). Opinion-driven communities' detection. *International Journal of Web Information Systems, 10(4)*, 324–342.

Dinsoreanu, M. B. (2012). Towards a semantic-driven automatic staging area design for heterogeneous data integration. *Proceedings of the 14th International Conference on Information Integration and Web-based Applications and Services* (pp. 290-293). ACM.

Dinsoreanu, M. P. (2013). Towards a Unified Thematic Model for Recommending Context-Sensitive Content. *Knowledge Discovery, Knowledge Engineering and Knowledge Management*, 68-83.

Esuli, A. S. (2006). Sentiwordnet: A publicly available lexical resource for opinion mining. *Proceedings of LREC* , (pp. 417-422).

Fang, F. K. (2014). Domain Adaptation for Sentiment Classification in Light of Multiple Sources. *INFORMS Journal on Computing* .

Garcia-Molina, H., Koutrika, G., & Parameswaran, A. (2011). Information seek-ing: convergence of search, recommendations and advertising. *Communications of the ACM, 54*(11), 121-130.

Ghosh., A. S. (2003). Cluster ensembles-a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 583-617.

Glorot, X. A. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, (pp. 513-520).

Go, A. B. (2009). *Twitter sentiment classification using distant supervision.* Stanford.

Griffiths, T. L., & Steyvers, M. (2002). A probabilistic approach to semantic representation. *The 24th Annual Conference of the Cognitive Science Society*, (pp. 381-386). Fairfax, Virginia.

Guang Qiu, B. L. (2011). Opinion Word Expansion and Target Extraction through Double Propagation. *Computational Linguistics, Vol. 37, No. 1*, 9-27.

Hao, F. (2005). *Investigating a heterogeneous data integration approach for data warehousing.* London: University of London.

Hasna O.L., M. F. (2014). Sentiment Polarity Extension for Context-Sensitive Recommender Systems. *Proceedings of International Conference on Knowledge Discovery and Information Retrieval (KDIR 2014)*, (pp. 126-137).

Heinrich, G. (2009). *Parameter estimation for text analysis.* Darmstadt, Germany: Fraunhofer IGD.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004, January). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems, 1*(22), 5-53.

Hu, M. &. (2004). Mining and Summarizing Customer Reviews. *Proceedings of the ACM SIGKDD International Conference on Knowledge on Knowledge Discovery and Data Mining (KDD-2004)*, (pp. 168-177).

Jiang, J. J. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*.

Kavitha, C., Sadasivam, G. S., & Shenoy, N. (2011). Ontology based semantic integration of heterogeneous databases. *European Journal of Scientifc Research*, 115-122.

L. Danon, A. D.-G. (2005). Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*.

L. Tang, X. W. (2010). *Community detection in multi-dimensional networks.* Arizona State University Tempe.

Lesk., M. (1986.). Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. *Proceedings of the 5th annual international conference on Systems documentation* (pp. 24-26). ACM.

Li, S. Z. (2008). Multi-domain adaptation for sentiment classification: Using multiple classifier combining methods. *In International Conference on Natural Language Processing and Knowledge Engineering* (pp. 1-8). IEEE.

Lin, C., He, Y., Everson, R., & Rüger, S. (2012). Weakly-supervised joint sentimenttopic. *IEEE Transactions on Knowledge and Data Engineering, 24*(6), 1134-1145.

Liu, B. (2010). Sentiment analysis and subjectivity. In N. I. Damerau, *Handbook of natural language processing, 2* (pp. 627-666).

Liu, B. (2012). *Sentiment Analysis and Opinion Mining.* Morgan & Claypool Publishers.

Liu, K. L. (2012). Emoticon Smoothed Language Models for Twitter Sentiment Analysis. *Proceedings of the 26th AAAI conference on Artificial Intelligence.* AAAI.

Lu, B. (2010). Identifying Opinion Holders and Targets with Dependency Parser in Chinese News Texts. *Proceedings of the NAACL HLT 2010 Student Research Workshop*, (pp. 46–51). Los Angeles, California.

M. Tsytsarau, T. P. (2011). Scalable detection of sentiment-based contradictions. *DiversiWeb*.

M.-C. de Marneffe, A. N. (2008). Finding contradictions in text. *Proceedings of the Association for Computational Linguistics*, 1039-1047.

Manning, C. D., Raghavan, P., & Schtze, H. (2008). *Introduction to Information Retrieval.* New York, NY, USA: Cambridge University Press.

Mansour, R. N.-H. (2013). Revisiting The Old Kitchen Sink: Do We Need Sentiment Domain Adaptation? *RANLP*, 420-427.

Martelot E. L., H. C. (2013). Fast multi-scale detection of relevant communities in large-scale networks. *The Computer Journal*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM, 38(11)*, 39-41.

Mohammad, S. M. (2013). Crowdsourcing a word–emotion association lexicon. *Computational Intelligence 29, no. 3*, 436-465.

Mohammad, S. M. (2013). NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. *arXiv preprint arXiv:1308.6242.*

Nakov, P. K. (2013). *Semeval-2013 task 2: Sentiment analysis in twitter.*

Neviarouskaya, A. H. (2011). Affect analysis model: novel rule-based approach to affect sensing from text. *Natural Language Engineering 17*, 95-135.

Nielsen, F. Å. (2011). *A new ANEW: Evaluation of a word list for sentiment analysis in microblogs.* arXiv preprint arXiv:1103.2903.

Nikulin, M. (2011). Hellinger distance. *Encyclopedia of Mathematics*.

Ortega R., F. A. (2013). SSA-UO: Unsupervised Twitter Sentiment Analysis. *SemEval 2013.*

Owoputi, O. O. (2013). Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters. *HLT-NAACL*, 380-390.

Paltoglou, G. a. (2012). Twitter, MySpace, Digg: Unsupervised sentiment analysis in social media. *ACM Transactions on Intelligent Systems and Technology (TIST) 3*, 66.

Pang, B. &. (2004). A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. *Proceedings of ACL*, (pp. 271-278). Barcelona.

Parrott, W. G. (2001). *Emotions in social psychology: Essential readings.* Psychology Press.

Patwardhan. (2003). *Incorporating dictionary and corpus information into a Context Vector Measure of Semantic Relatedness.* Duluth: University of Minnesota.

Phan, X. H. (2013, March 20). *Jgibblda: A java implementation of latent dirichlet allocation (lda) using gibbs sampling for parameter estimation and inference.* Retrieved March 20, 2013, from http://jgibblda.sourceforge.net.: http://jgibblda.sourceforge.net.

Plutchik, R. (2001). The nature of emotions. *American Scientist 89, no. 4* , 344-350.

Raaijmakers, S., & Kraaij, W. (2010). Classifier Calibration for Multi-Domain Sentiment Classification. *Proceedings of the 4th ICWSM*.

Reporting verbs. (n.d.). Retrieved 2014, from http://www.adelaide.edu.au/writingcentre/learning_guides/learningGuide_reportingVerbs.pdf.

Ribeiro-Neto, B., Cristo, M., Golgher, P. B., & Silva de Moura, E. (2005). Impedance coupling in content-targeted advertising. *28th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 496-503). Salvador, Brazil: ACM.

Rosvall M., B. C. (2011). Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems. *PloS one, 6(4):e18209*.

Russu, R. M. (2014). An opinion mining approach for Romanian language. *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on* (pp. 43-46). IEEE.

S. Harabagiu, A. H. (2006). Negation, contrast and contradiction in text processing. *AAAI* , 755-762.

Saif, H. F. (2013). Evaluation datasets for twitter sentiment analysis. *Proceedings, 1st Workshop on Emotion and Sentiment in Social and Expressive Media (ESSEM) in Conjunction with AI*IA Conference.* Turin.

Santos, R. L., Macdonald, C., & Ounis, I. (2010). Exploiting query reformulations for web search result diversification. *19th international conference on World wide web* (pp. 881-890). Raleigh, North Carolina, USA: ACM.

Skoutas, D., Simitsis, A., & Sellis, T. (2009). Ontology-driven conceptual design of etl processes using graph transformations. *Springer Journal on Data Semantics (JoDS)*, 119-145.

Speriosu, M. N. (2011). Twitter polarity classification with label propagation over lexical links and the follower graph. *Proceedings of the First workshop on Unsupervised Learning in NLP* (pp. 53-63). ACL.

Suciu D.A., I. V. (2014). Learning Good Opinions from Just Two Words Is Not Bad. *Proceedings of International Conference on Knowledge Discovery and Information Retrieval (KDIR 2014)* (pp. 233 – 241). ScitePress.

Taboada M., A. C. (2006). Methods for Creating Semantic Orientation Dictionaries. *Proceedings of 5th International Conference on Language Resources and Evaluation (LREC)*, (pp. 427-432).

The Economist. (2010). *The data deluge*. Retrieved April 22, 2012, from http://www.economist.com/node/15579717

Thelwall, M. B. (2010). Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology, 61(12)*, 2544–2558.

Thelwall, M. B. (2012). Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology, 63(1)*, 163-173.

Turney, P. (2002). Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, (pp. 417-424).

V. Ionescu, &. M. (2013). An approach to mining financial markets through market state classification. *2013 IEEE 9th International Conference on Intelligent Computer Communication and Processing (ICCP)* (pp. 43-46). IEEE.

Wilson T., K. Z. (2013). SemEval-2013 task 2: Sentiment analysis in twitter. *Proceedings of the International Workshop on Semantic Evaluation.*

Winkler, W. E. (1990). String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. *Proceedings of the Section on Survey Research Methods* (pp. 354-359). American Statistical Association.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques* (3rd ed.). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Xia, R. C. (2013). Feature ensemble plus sample selection: domain adaptation for sentiment classification. *Intelligent Systems, IEEE 28, no. 3* , 10-18.

Xia, R., Zong, C., & Li, S. (2011). Ensemble of feature sets and classification algorithms for sentiment classification. *Information Sciences, 181*(6), 1138-1152.

Yih, W.-t., Goodman, J., & Carvalho, V. R. (2006). Finding advertising keywords on web pages. *15th international conference on World Wide Web* (pp. 213-222). Edinburgh, Scotland: ACM.

Zhang, Y., Surendran, A. C., Platt, J. C., & Narasimhan, M. (2008). Learning from multi-topic web documents for contextual advertisement. *14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1051-1059). Las Vegas, Nevada, USA: ACM.

Zieg, P. (2007). *The SIRUP approach to personal semantic data integration.* Zurich: University of Zurich.

Ziegler, C.-N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005). Improving recommendation lists through topic diversification. *The 14th International Conference on World Wide Web* (pp. 22-32). Chiba, Japan: ACM.