Department of Computer Science
Series of Publications A
Report A-2018-7

Classification and clustering in media monitoring: from knowledge engineering to deep learning

Lidia Pivovarova

To be presented, with the permission of the Faculty of Science of the University of Helsinki, for public criticism in Auditorium D122, Exactum building, on December 21st, 2018, at 12 o'clock noon.

University of Helsinki
Finland

Supervisor
University Researcher Roman Yangarber, University of Helsinki, Finland

Pre-examiners
Professor Robert Gaizauskas, University of Sheffield, UK
Dr Jussi Karlgren, KTH Royal Institute of Technology, Sweden

Opponent
Professor Heng Ji, Rensselaer Polytechnic Institute, USA

Custos
Professor Jyrki Kivinen, University of Helsinki, Finland

Contact information

Department of Computer Science
P.O. Box 68 (Pietari Kalmin katu 5)
FI-00014 University of Helsinki
Finland

Email address: info@cs.helsinki.fi
URL: http://cs.helsinki.fi/
Telephone: +358 2941 911

ISSN 1238-8645
ISBN 978-951-51-4700-4 (paperback)
ISBN 978-951-51-4701-1 (PDF)
Computing Reviews (1998) Classification: I.2.7, H.3.3
Helsinki 2018
Unigrafia

Classification and clustering in media monitoring: from knowledge engineering to deep learning

Lidia Pivovarova

Department of Computer Science
P.O. Box 68, FI-00014 University of Helsinki, Finland
lidia.pivovarova@cs.helsinki.fi
https://www.cs.helsinki.fi/people/pivovaro

PhD Thesis, Series of Publications A, Report A-2018-7
Helsinki, December 2018, 78+56 pages
ISSN 1238-8645
ISBN 978-951-51-4700-4 (paperback)
ISBN 978-951-51-4701-1 (PDF)

Abstract

This thesis addresses information extraction from financial news for decision support in the business domain. News is an important source of information for business decision makers, which reflects investors' expectations and affects companies' reputations. A vast amount of various news sources forces development of text mining algorithms to collect most crucial information and present to a user in a condensed form.

The thesis presents the PULS media monitoring system and describes several news mining tasks, namely document clustering, multi-label news classification and text polarity detection. For each task, we present an end-to-end processing pipeline, starting from data preprocessing and clean-up. A particular attention is given to named entities (NEs), that are used as one of the inputs for all presented algorithms.

Chapter 1 overviews the PULS news monitoring system and its niche within text mining for business intelligence.

In Chapter 2 we propose a novel algorithm for news grouping, which uses NE salience and exploits a specific structure of news articles.

In Chapter 3 we use automatically extracted NEs and entity descriptors in combination with keywords to improve SVM classifiers for large-scale multi-label text classification. Then, we propose a convolutional neural network (CNN) architecture that outperforms an ensemble of SVM classifiers for two different datasets. We compared various ways to represent NEs for CNN classifiers.

In Chapter 4 we use a CNN classifier for entity-level business polarity detection. We compare three methods of re-using data annotated for a different though remotely related task and demonstrate that unsupervised knowledge transfer works better than other techniques that involve manual mapping.

The thesis concludes with a summary of the main findings of the work. The key findings presented in the thesis have been published in top NLP conferences, including EACL and NAACL. The thesis also presents additional experiments that have not been published previously.

Computing Reviews (1998) Categories and Subject Descriptors:
I.2.7 [Artificial Intelligence]: Natural Language Processing - Text analysis
H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms:
Natural Language Processing, Machine Learning

Additional Key Words and Phrases:
Text Classification, Deep Learning, Knowledge Engineering, Business Intelligence, Media Monitoring

Acknowledgements

I am indebted to my supervisor, Dr Roman Yangarber, who invited me to work in the PULS project and gave me countless opportunities to learn various exiting topics and to put into practice my own ideas.

This work could not be done without my wonderful colleagues, who worked in Roman's group at various times: Mian Du, Llorenç Escoter i Torres, José María Hoya Quecedo, Silja Huttunen, Anisia Katinskaya, Javad Nouri, Mikhail Novikov, Natalia Ostapuk, Matthew Pierce, Peter von Etter, and Han Xiao.

I am very grateful to Prof. Arto Klami, co-author in some of my recent work, which constitutes the deep learning parts of this thesis.

I very much appreciate the thorough work done by the pre-examiners, Prof. Robert Gaizauskas and Dr Jussi Karlgren, which helped me to convey my thoughts in a more comprehensible form.

I am thankful to the Finnish Center for International Mobility (CIMO)—this work started with their scholarship. Many thanks to Dr Mikhail Kopotev, who encouraged me to apply for the scholarship.

I am grateful to the Doctoral Programme in Computer Science (DoCS) and the Helsinki Doctoral Programme in Computer Science (Hecse), and to Dr Pirjo Moen in particular.

The work was supported in part by Tekes (the Finnish Funding Agency for Technology and Innovation), Project *iMEDL: Digital Media Evolution through Deep Learning* (Grant number 5933/31/2017); Academy of Finland Center of Excellence "ALGODAN": Algorithmic Data Analysis; Tekes Project: *PULS Information Platform*; Stockholm Environment Institute (SEI), Sweden, Project: *Automatic Data Gathering for the Arctic Resilience Report*; Frontex: European Border and Coast Guard Agency Project: *Capturing Structured Information on Illegal Immigration Events, Cross-border Criminal Activities and Related Crisis Events from Online News.*

I dedicate this work to the memory of my first advisor Prof. Valery Rubashkin, who gave my career the right start.

Helsinki, December 2018
Lidia Pivovarova

Contents

Chapter 1

Introduction

1.1 PULS news monitoring system

This thesis summarizes the author's work in the PULS project aimed at news gathering, processing, verification and visualization. PULS[1] is a media monitoring system, which has been applied to several domains, including Cross-Border Security [3, 4, 5, 67] and Epidemic Surveillance [38, 39, 40, 55]. This thesis is focused on PULS application to business intelligence. In the business domain PULS aimed at processing large amounts of financial news to extract main events and monitor company activities.

PULS daily collects about 10000 articles from more than 1000 news sources related to business. The documents are then processed using a natural language processing (NLP) pipeline, which extracts named entities and events from the text [39, 43, 44, 70]. Each document is then classified by a set of classifiers that define document relevance [41, 42], assign industry and topic labels [24, 25], and find polarity of company mentions [69, 71]. Finally, documents are grouped in an unsupervised fashion, with each group representing a particular story [30]. Names and classified labels are then aggregated across the group, and outliers that are found only in a small percentage of the group documents are discarded. These groups are presented in the user interface, which provides the user a quick overview of the current business events and various possibilities to further search and investigate news stories [26].

PULS core NLP engine is an Information Extraction (IE) system that works using a cascade of patterns, starting from low-level patterns that match multi-word

[1] http://puls.cs.helsinki.fi

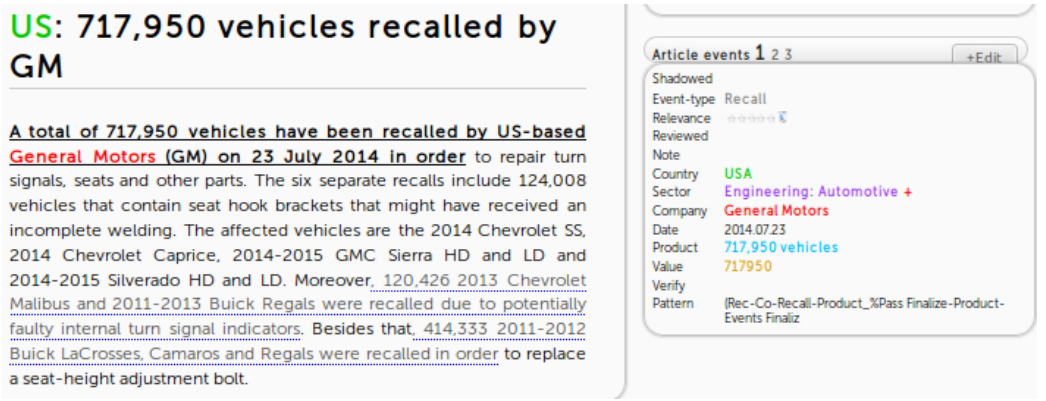


Figure 1.1: Component of the user interface: input document and RECALL event extracted by PULS.

units in text (e.g., noun groups or complex digits), then named entity recognition patterns and reference resolution rules, and finally high-level patterns that match complex events such as, in the business domain, company mergers or product recalls. An example event, extracted by the PULS system is shown in Figure 1.1. The text mentions a product recall conducted by General Motors. The predefined template contains several slots, such as company name, product, date, location and so on. These slots are filled with the information from text using event-specific patterns.

The goal of an IE system is to transform unstructured text into a set of structured records, which can be stored in the database, indexed, searched and analyzed using standard data mining techniques. In that sense PULS can be seen as a standard IE system that works similar to other systems of that type (there are several overviews of traditional IE systems [1, 16, 68]). The main strength of PULS is in the amount of knowledge that has been collected over the years, including knowledge bases, dictionaries, thousands of patterns and rules. Some of these resources were created manually and some were collected using pattern mining techniques [89, 90, 91].

A vast amount of underlying knowledge allows PULS to analyze text with high accuracy. However, the knowledge-based approach has its drawbacks. The main problem is that an IE system usually extracts only those facts that are specified beforehand in its templates. Adding a new type of facts to the system takes significant effort, since that requires construction of a new set of patterns, which could be quite large because the same fact may be expressed in text in various ways.

This leads to a growing interest in open information extraction, i.e., extraction of all possible facts from the text without specification of templates [6, 11, 61]. This approach is quite promising for the population of knowledge bases but rather questionable for domain-specific systems, such as news monitoring for the business domain. There is a huge variety of activities in which companies could be involved but that does not mean that all random facts that can be found in financial news are equally valuable for the end user.

PULS started out as a relatively simple IE system that was applied to small well-defined scenarios, such as to figure out who got sick with what disease. Then the problems got much bigger, as the scenarios changed and the users changed and their demands grew. PULS has evolved for over 15 year, adapting to new use-cases, to different user bases and requirements, with the author of this thesis being involved for approximately the last 7 years.

Over these 15 year major changes have happened in automatic text processing. The amount of easily available data has increased enormously with the growth of the Internet and mobile technologies. New types of texts, e.g., blogs, microblogs, text messages and chats attract researchers' attention and produce new areas of research. During the same time, machine learning techniques were developing and computational powers were growing. The cumulative effect of these two processes produced impressive results and came to the recent boost of deep learning that completely changed the research agenda in the area of text processing.

All these changes affected the PULS project, as we were able to solve increasingly complex problems more effectively. Over the last several years the PULS project has switched from single-document pattern-based IE to more complex analysis of the news stream as a whole. A fragment of the newest user interface with document groups is presented in Figure 1.2. The groups in this screenshot are sorted by date and it can be seen that most of the events are represented by a single document, though some more important events are reported in several news sources, such as the Equifax data breach (in the top right corner). The user interface also allows sorting events by size and thus shows the biggest news stories over the last several days.

All information presented in the group view in Figure 1.2 is extracted from documents automatically. The most salient names extracted by the Named Entity Recognition (NER) module are shown in the top bar of each group; name size represents its salience for the group. Green color means the event is positive for this company, red color means a negative event. Other information for the group includes topic and industry labels ("Data Protection: Freedom of Information" and "I-Finance" for the Equifax group).

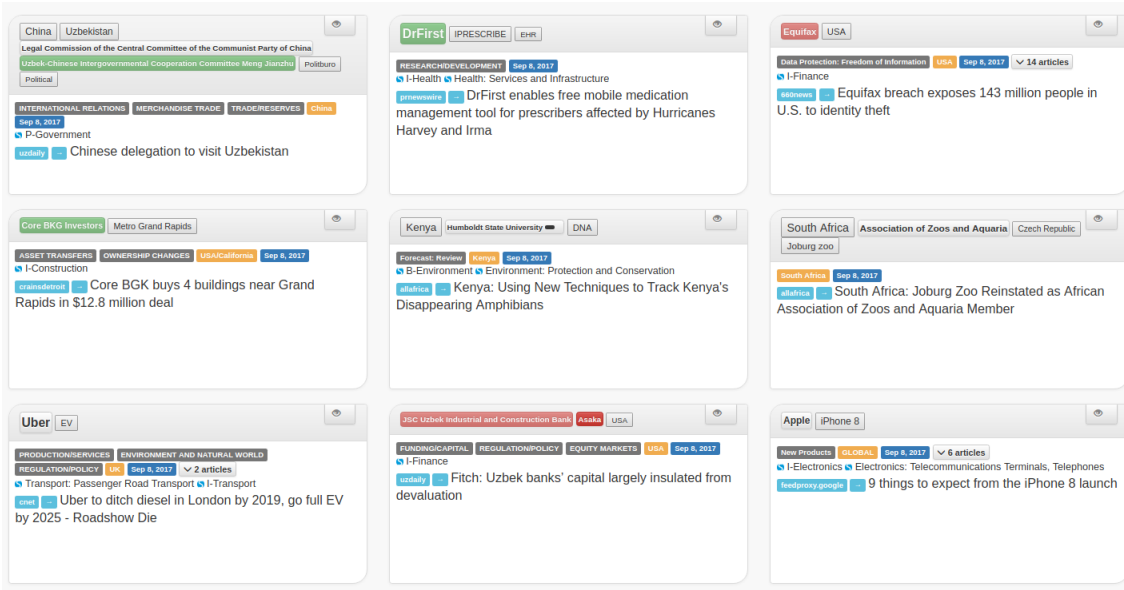


Figure 1.2: Group view.

The text in the group box is a headline of the most relevant document. The group can be expanded as presented in Figure 1.3 with the list of the group headlines. The number next to each headline is a cosine distance between this headline and the medoid one. Finally, by clicking on a headline the user can read a short summary of a document as shown in Figure 1.4.

News stream representation in the form of document groups, shown in Figure 1.2, does not follow any particular structure—the system can find as many names or topic labels as are relevant to the event. This representation gives the end-user an instant insight into the content of a document group together with number of options to investigate it further, to read an event summary and to search other events involving the same names.

Thus, it is doubtful whether the user still needs structured information in the form of table slots, exemplified in Figure 1.1. As has been said before, a tuning system to produce this representation is time-consuming and can be done only for a limited number of event types specified beforehand. In contrast, a group-based event representation can be built for any set of documents even if a particular event has never been seen before and has no corresponding template and patterns. It is also less vulnerable to an idiosyncrasy of a certain text since important events

Figure 1.3: Group headlines.

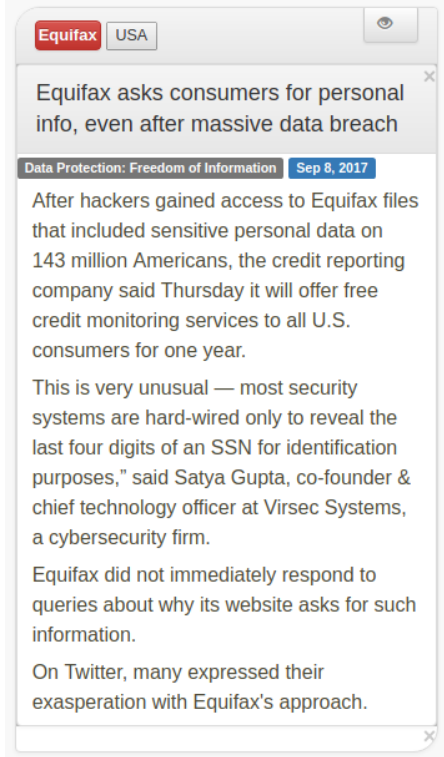


Figure 1.4: Document summary.

are re-printed in many sources. Even if the system makes a mistake parsing a particular document, the group representation for the main events is usually correct.

This does not mean that the PULS resources collected over the years became useless. On the contrary, the PULS IE system is a core component, which underlies produces features for other components. Even when a component employs a standard ML technique, it has some advantage, since it uses linguistically-motivated features. For example, it was shown that usage of proper names and background information collected for each name over the years by the IE system improves performance of the sector classification [25].

In that sense PULS approach is different from the current mainstream in NLP, which tends to neglect a linguistic analysis. For example, the common practice for building word vectors is to use all tokens in a corpus as they are, without normalization. This might be partially explained by the fact that most of these works are done for the English language. However, even English has such phenomena as phrasal verbs and conjugations and a variety of accurate normalization tools already exists. The general approach of the PULS project is to utilize as much linguistic knowledge as possible and to use machine learning techniques in combination with knowledge-based analysis.

1.2 Text Mining for Business Intelligence

Business intelligence is a set of analysis techniques aimed at decision support in the area of business. Different data sources are important in business intelligence. The main distinction lies between technical and fundamental data [64], i.e., between structured quantitative data, such as historical market prices or company indicators, and unstructured data, such as company reports and announcement, customer feedback, news feeds, social media. The growing need in processing unstructured data caused interest in applying text mining techniques for the business domain: "around 85% of all business information exists as unstructured data, while 60% of CIOs and CTOs considered unstructured data as vital for improving procedures and creating new business opportunities"[79].

Text mining has a number of applications in the business domain: stocks and Forex (foreign exchange) predictions; customer relation management including opinion mining and customer feedback analysis; cyber-security, including anti-phishing and anti-spam, fraud and malware detection; knowledge mining, including automatic ontology and knowledge base construction; analysis of company

annual reports and disclosures, including readability studies; facilitation of information retrieval and knowledge organizing for business professionals [32, 54].

Documents used in business intelligence are of various types. They can be produced by companies themselves, e.g., annual reports, press releases or letters to shareholders, or by external sources, e.g., news, social media or analytic reports [32, 54]. External information about market players is crucial for business decisions and may affect investors' behavior even when it is not supported by changes in company fundamentals [37].

News is an important source of information for business decision makers. Financial news and markets are mutually dependent [51], with news playing the role of a "magnifying glass," which makes economic growths and declines more visible [86]. Even though news does not contain new information about fundamental asset values (such information can be found in technical sources), they reflect investor sentiment [82].

Moreover, news may cause a long-term effect on company reputation and thus affect its performance [51]. At the same time, news sources have their own reputation and vary in customers' trust. "Financial news ... is often partly unreliable, one-sided, unbalanced, inconsistent with prior knowledge, preliminary, outdated, PR spin and at the very least incomplete" [51]. There are evidences that markets are affected by representation of events in the media and different news sources may cause different effects [29]. Thus, it is necessary to take into account as many various sources as it is possible and have the possibility to compare and verify information that comes through different channels, which nowadays can be done automatically or semi-automatically by means of news aggregating systems. However, many studies that investigate the relation between media and market behavior depend on some particular news source, such as Wall Street Journal [82] or Dow Jones Newswire [9].

The growing amount of information and the increasing capacity for faster processing has compelled large companies to adopt automatic data processing techniques [37]. However, text mining techniques applied in business are frequently out-dated. As far as can be judged from literature, advanced developments in NLP are rarely used in this domain. Authors of the recent survey [32] reviewed 266 papers on text mining in financial domains. They demonstrate that out of 266 papers 86 describe manual text analysis including dozens of studies published in the 2000s. A total of 81 employ "basic text mining," which in authors' terminology means automatic text analysis without ML. According to the study, the number of these "basic" studies do not decrease and is approximately equal to ones using ML. Moreover, the same survey demonstrate that among ML techniques the

vast majority is taken by well-established methods such as SVM (support-vector machine) classifiers. Only 8 out of the 266 reviewed studies employ artificial neural networks (ANN), which contrasts with NLP literature, where deep learning became a mainstream over the last five years. Similarly, only 7 studies employ matrix decomposition techniques such as LSA (latent semantic analysis) or SVD (singular-value decomposition), even though these methods have a longer history in NLP than neural networks.

The same situation is presented in another survey of text mining applications in the financial domain [54]. 75% the reviewed papers use "regular classification methods" such as SVMs, Naive Bayes, linear regression or k-NN (k-nearest neighbors). Only a dozen of the reviewed papers employed neural networks, and only a couple of papers found for SVD and hidden Markov models. In the introductory part the authors of this review state that text analysis always starts with construction of a term-document matrix, thus completely ignoring more recent structured models or continuous data representation.

Note that more elaborate ML techniques such as times series prediction and pattern mining are widely used in business intelligence, e.g., for stock price prediction or fraud detection, but their use is limited by technical, i.e., structured, data. See, for example, an overview of ANN applications in the business domain [83], which focuses entirely on structured data even for such a task as customer satisfaction studies.

The PULS media monitoring system aims to bridge the gap, described above, between a growing need in automatic processing of large news streams and the current state of development in this domain. PULS processes thousands of news articles daily and presents them in a set of summary views that allow the end user to quickly grasp main events of the day, most often mentioned companies, and media attitude towards them. PULS uses a range of text mining techniques, ranging from linguistically-motivated rules to modern deep-learning methods. In this thesis we describe in details several PULS components and demonstrate that combination of knowledge-based techniques with advanced ML approaches yields the best results than any of these methods alone.

From user point of view PULS can be used in various scenarios. In the most general, exploratory mode, PULS can be used as a news aggregator. In that case the user quickly scans the most important news of the day. The user can also constrain the system output using sets of labels and read only news related to certain location, industry sector or event type. E.g., it is possible to find all product launches in Finland or read only news about retail in the US.

PULS can also be used as a decision support system. In that case the user investigates a company profile, that presents, in a condensed form, all mentions of the company in the news. Using the profile the user can find the most frequent locations, where the company operates; its biggest partners; whether the company usually mentioned in positive or negative context. It is also possible to compare the company with similar companies, i.e. companies that are described similarly in the news.

These are common use cases that require of the system a deeper semantic understanding of the data, including the need for entity recognition and linking, sentiment analysis, text classification and clustering, etc. These use cases are not adequately served by the "standard" computational tools at a user's disposal, namely keyword-based search engines.

1.3 Thesis overview

The author of this thesis has been involved in the development of the PULS news monitoring system for a number of years being responsible for various tasks, including maintenance of dictionaries, knowledge bases and lexical resources, development and implementation of text classification and clustering algorithms, data annotation and experiments. The thesis summarizes several of the most recent papers on PULS business intelligence. The papers are grouped into several topics, each topic forms one chapter of the thesis and comprises one or more papers. The list of papers with the author's contribution is presented below.

1.3.1 Chapter 2: News grouping

Paper I

Llorenc Escoter, Lidia Pivovarova, Mian Du, Anisia Katinskaia, Roman Yangarber *Grouping business news stories based on salience of named entities* 15th Conference of the European Chapter of the Association for Computational Linguistics Proceedings of Conference (EACL). 2017.

In this paper, I am one of the two primary authors. I proposed the idea to compute named entity salience as a combination of its frequency and prominence. I implemented the first version of the manual grouping interface, which uses several heuristics to minimize a number of documents presented to an annotator and thus to reduce efforts spent on the annotation. The code was used to annotate a test set, I annotated a substantial portion of the data myself. I made major contributions

to the experimental design, analysis of the results, the text and the structure of the paper.

1.3.2 Chapter 3: Multi-label text classification

Paper II

Mian Du, Matthew Pierce, Lidia Pivovarova, Roman Yangarber *Supervised Classification Using Balanced Training* International Conference on Statistical Language and Speech Processing (SLSP). 2014.

I made major contributions to the experimental design, analysis of the results, the text and the structure of this paper.

Paper III

Mian Du, Matthew Pierce, Lidia Pivovarova, Roman Yangarber *Improving Supervised Classification Using Information Extraction* International Conference on Applications of Natural Language to Information Systems (NLDB). 2015.

I was responsible for the IE system, which was used to obtain features for the experiments, described in this paper. I made major contributions to the experimental design, visualization of the data, analysis of the results, the text and the structure of this paper.

Note: This chapter also presents experiments with convolutional neural networks (CNN). I implemented most of the code and conducted most of these experiments myself.

1.3.3 Chapter 4: Business polarity detection

Paper IV

Lidia Pivovarova, Arto Klami, Roman Yangarber *Benchmarks and models for entity-oriented polarity detection* The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL/HLT). 2018.

I implemented most of the code and conducted all experiments for this paper. I made major contributions to the the text and the structure of this paper.

Chapter 2

News grouping

2.1 Named Entity Salience

Entity *salience* is a measure of the entity importance in the content of the document. "Salience is a function of the structure of a text, and indirectly a function of the intention of the author, as opposed to a function of the reader's intent or needs" [36]. Thus, detection of salient entities is similar to the task of determining the *aboutness* of the document, though the latter has a broader meaning: aboutness may be represented by keywords, tags from a fixed dictionary, main topics, or short summaries [36, 74]. Aboutness, based solely on lexical elements—such as keywords or summaries—may be incomplete, redundant or misleading, that is why it is crucial to build a semantic representation, i.e., by means of elements that are not implicitly presented in the text. Named entities (NEs) are one example of such semantic elements and they are not equal to proper names, since an entity may have different surface forms within the document or across the corpus.

The PULS NER module recognizes proper names in text, defines their type—COMPANY, PERSON, LOCATION, PRODUCT, or UNSPECIFIED—and merges variants of the same name, e.g., "Walmart", "Wal-Mart", and "Wal-Mart Stores, Inc.". PULS NER is based on cascade of patterns that match noun groups. In addition to name, these pattern find *entity descriptors*, such as "retail chain" or "multinational retail corporation". Name variants and descriptors for each company are accumulated in the PULS knowledge base and form the company profile. Since the NER module merges various mentions of an entity within a document we can determine the entity salience based on its frequency and position in the text.

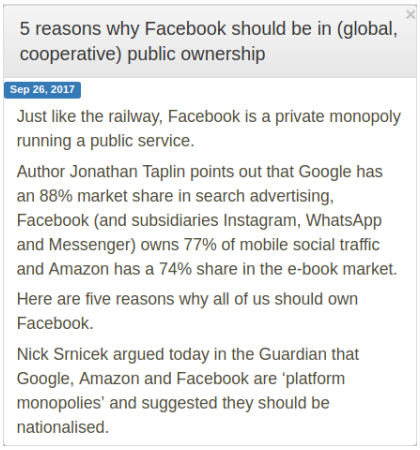
The idea that entities are different in salience has long been established in cognitive studies; "objects with high salience are the focus of attention; those with

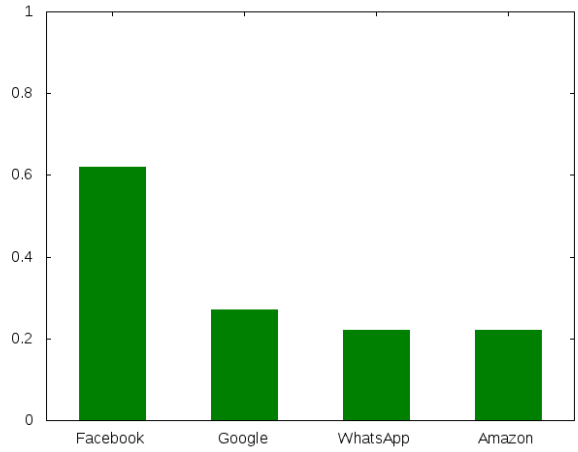
low salience are at the periphery" [8]. However, empirical studies in this area are still rare. Salient scores are frequently used in text summarization [35], but rarely in text clustering, which is our primary task in this chapter. Named entities can be used as features for clustering [62], though they are usually weighted according to TF-IDF [84] or its variants [12, 50], i.e., by statistics based on bag-of-words representation that completely ignores document structure.

There are few datasets for named entity salience detection. Two of them, namely [22, 85] consist of only a few hundred documents, since salience annotation is a difficult and time-consuming task. Another corpus [28] consists of more than 2 million documents and was annotated using a semi-automatic procedure was used: the authors used a New York Times corpus [77], where each article has a manually written abstract; entities mentioned in the abstract are considered to be salient. Thus, although salience was determined automatically, time-consuming manual work had been done in the earlier stage. Moreover, this approach allows only to classify entities as salient or non-salient, without any ranking of entities. In other works that involved direct manual annotation of the entities they were categorized in four or three classes of salience, though it was observed that human annotators are much better at finding top-salient entities than at distinguishing among more subtle classes [22, 28].

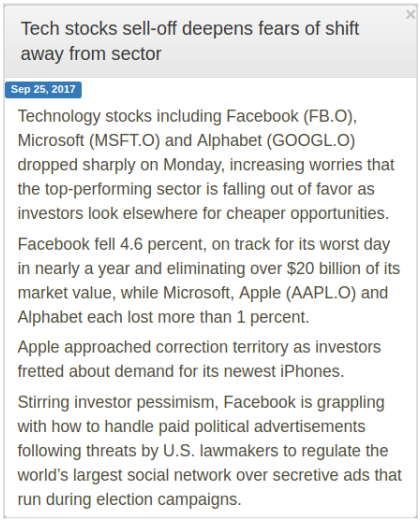
In PULS entity salience is determined in an unsupervised fashion and is defined as a real number between 0 (non-salient) and 1 (most salient). Thus, instead of simply classifying entities into salient and non-salient, we use a vector representation of the document where all entities are taken into account, each with its own salience weight. For example, in Figure 2.1 two documents are represented by salience histograms: the first one talks mostly about Facebook and mentions some other companies for background information; the second document gives almost equal attention to several companies. Although Facebook is the most salient company in both documents, the stories are completely different, which can be seen directly in the histograms, without reading the texts.

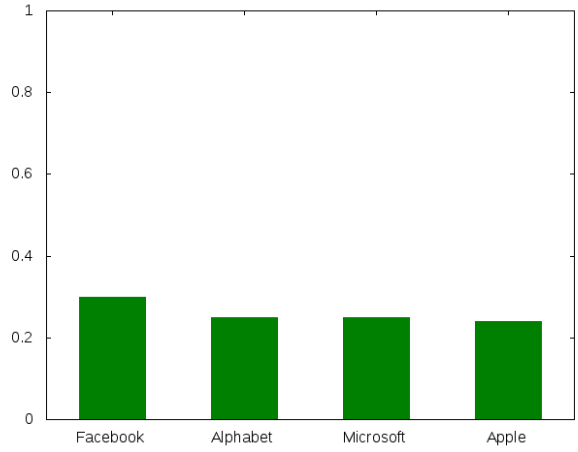
A particular formula for the salience exploits the specific structure of news articles. Authors often strive to mention the main event in the title in a condensed form, then the same information elaborated in the first few sentences, followed by greater detail and background. Thus, the most important NEs are mentioned early in the text and then repeated, while less relevant NEs are mentioned in the lower paragraphs and are less frequent. This observation is frequently utilized in salience detection tasks, where name position and count is used as the most important features [22, 28, 85]; this was also shown for another type of texts, namely web-pages [36].





(a)





(b)

Figure 2.1: Two documents and corresponding salience histograms.

In the PULS news monitoring system *salience* is computed as a combination of *prominence* and *frequency* of an entity in a document. Prominence captures the

importance of the first mention of entity e in document d:

$$\textit{prominence}(e,d) = \frac{\mathrm{ns}(d) - \mathrm{fs}(e,d)}{\mathrm{ns}(d)} \tag{2.1}$$

where $\mathrm{ns}(d)$ is the total number of sentences in the document, $\mathrm{fs}(e,d)$ is the sentence number, starting at zero, of the first mention of e in d. Thus, the prominence for entities mentioned in the title is 1. Prominence also takes into account the total length of the document, to capture diversity of news sources in the collection.

Frequency is the ratio of entity mentions over all NE mentions in the document:

$$\textit{frequency}(e,d) = \frac{C(e,d)}{\sum\limits_{e' \in \mathrm{NE}(d)} C(e',d)} \tag{2.2}$$

where $C(e,d)$ is the count of e in d, $NE(d)$ are all NEs in d. Note that we compute the NE frequency relative to the other NEs only and ignore all the other words in the document, since NEs and common words have rather different distributions: important terms are usually repeated more times than names.

Salience is defined as the geometric mean of prominence and frequency:

$$S(e,d) = \sqrt{\textit{prominence}(e,d) \cdot \textit{frequency}(e,d)} \tag{2.3}$$

Salience is between 0 and 1, but the saliences in a document need not add up to one—there may be more than one salient entity in the document, or none.

We make extensive use of salience in our system to aggregate and present information to users. For example, for each group of related documents we present a list of salient companies, where the least salient companies are hidden. Similarly, when a user searches for a name, the system returns only documents where entity salience is above a certain threshold. Another application is text summarization, where NE salience is used to extract the most important sentences in the text. Since manual annotation of entity saliences in a document is a time-consuming task and it is difficult for humans to annotate with real values, we do not have at our disposal any dataset relevant to salience detection. Instead, we use indirect evaluation by means of other tasks, such as document grouping, described in the next section and published in Paper I.

2.2 Document grouping

2.2.1 Task Definition and Dataset

The role of the grouping module in the PULS news monitoring system is to find and cluster together documents that describe the same *story*, i.e., an event or a set of closely related events that involves the same main actors. Depending on the substance of the event, stories last in different time spans: some appear only once while others last for several weeks and even months. Currently PULS does not extract long-lasting stories; grouping is applied to documents within one day. We tried different time spans and empirically found that one-day span is optimal for business monitoring needs.

There are two primary goals for the grouping. The first goal is to minimize redundancy and ensure that the user sees a given story only once and is able to get all information related to the story in one place. The second goal is to find trending stories, i.e., stories that appeared in multiple sources. This task is different from text topic categorization, since grouping is fine-grained. NEs play a crucial role in the task definition, since stories may differ only in names of the involved actors. This is especially crucial for the business domain, where stories are stereotypical—contracts, investments, nominations and so on—and news frequently use formulaic language, with the same phrases and constructions repeated over and over again. As a consequence, we cannot use any standard news classification dataset, such as RCV1 or 20 Newsgroup, since they use more general classification scheme.

Thus, we annotated a novel dataset on the basis of our business corpus. We selected one "typical" day for annotation, with a total of almost 4000 documents[1]. Annotation was done by means of the command line interface, which displays a pair of documents and allows an annotator to make three main decisions: the pair should be grouped if the documents describe the same story; non-grouped, if the stories are different; partially grouped, if the stories are different but related in any sense. Then the groups are constructed on the basis of these annotations. The partial grouping was ignored in our experiments, i.e., partially grouped documents are considered to be non-grouped; however, this relation is stored in the dataset and can be used in future experiments.

The relations among documents are stored in a triangular matrix. Since in reality the size of this matrix is 4000 times 4000, manual annotation of each document pair is out of the question, this would require millions of operations.

[1]This typical day was collected in November 2015; since then the PULS list of news sources has been expanded currently we collect almost twice as many documents a day.

Thus, we implemented a number of devices to aim at reducing the annotators' efforts. First, we initialize the matrix by pre-making all document pairs that do not have a name in common as non-grouped. Another idea that allows to minimize human efforts is *decision propagation*, which means that (ideally) only one member of a given group should be checked against a member of another group. This process may sometimes cause contradictions. Contradictions may happen due to inaccuracies in NER, for example, if a name has different forms in different documents. In that case an annotator is warned and asked to resolve the contradiction.

The annotation process is illustrated in Figure 2.2, where we present a grouping matrix at various stages of annotation process; '+' means grouped documents, '–' means non-grouped, and empty cell means the relation is to be annotated. Figure 2.2a illustrates an amount of work that should have been done without initialization: most cells are empty and need to be filled manually. Figure 2.2b shows a matrix with initialization: many cells, i.e. document pairs, can be *apriori* excluded from manual annotation. Figure 2.2c exemplifies decision propagation: an annotator decides to group the third and the first documents and this decision is propagated and the 6th and 8th documents, un-grouped with the first, become un-grouped with the third. Figure 2.2d shows a contradiction: an annotator tries to group the first and the seventh documents, which contradicts initial decision to un-group 7th and 1st.

In total, 4 members of PULS team were involved in the annotation, which spanned across two calendar months. This was a huge investment of time for our team, so most of the documents were annotated by one person and cross-agreement cannot be computed for the dataset. However, in the beginning of the process we annotated several cases together and discussed difficult ones to work out general comprehension of what should be grouped. This task seems to be relatively easy for annotators, in majority of the cases it is obvious whether documents should be grouped together or not.

2.2.2 Evaluation

We use V-Measure [76] to evaluate the grouping algorithm performance. V-Measure is the harmonic mean of homogeneity (H) and completeness (C) [76].

$$H = 1 - \frac{\mathbb{H}(C|K)}{\mathbb{H}(C)} \qquad C = 1 - \frac{\mathbb{H}(K|C)}{\mathbb{H}(K)} \qquad V = \frac{2HC}{H + C} \tag{2.4}$$

where $\mathbb{H}$ denotes entropy, K are predicted labels, and C are the true labels. Com-

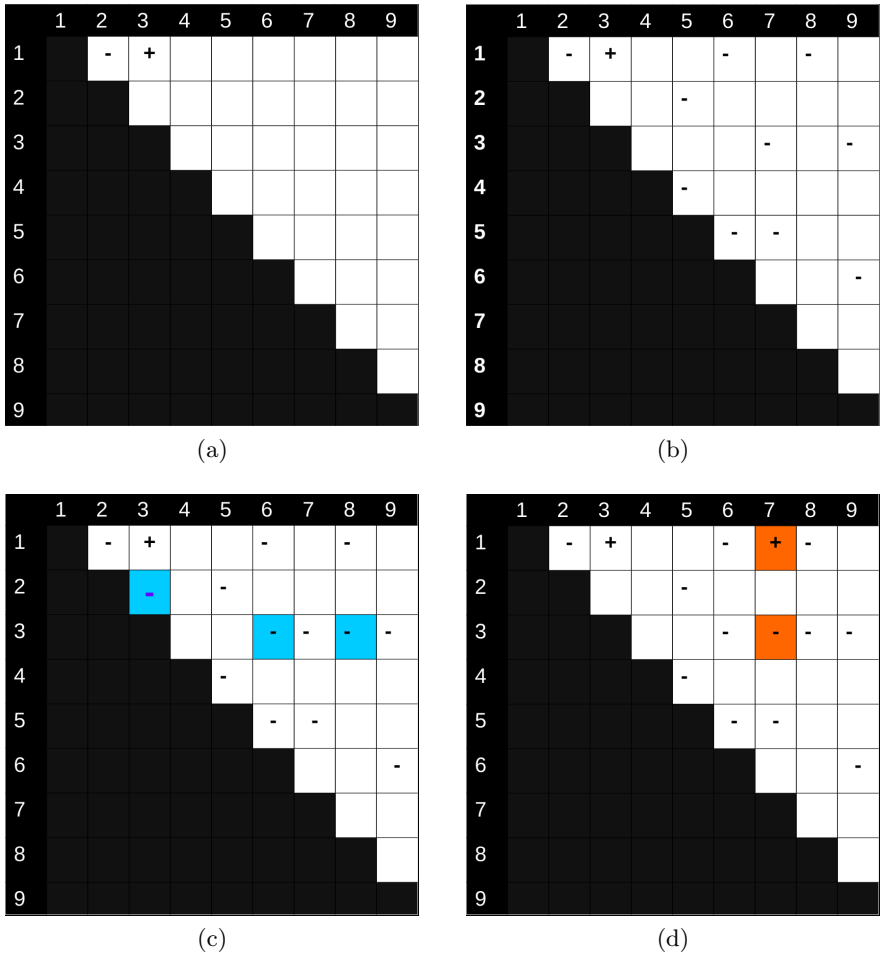


Figure 2.2: Relation representation for manual annotation.

pleteness, homogeneity and V-measure are analogous to recall, precision and F-measure respectively.

In Paper I we proposed a novel method to adjust V-measure against *naïve* strategy, i.e. against applying no grouping and assigning each document to its own cluster. This is a hard baseline, which yields (unadjasted) V-measure of 0.965, because the naïve strategy by definition has a perfect completeness. Moreover, a news stream contains many minor news that should not be grouped with any other document.

The adjustment is computed according to the following formula:

$$V_{\text{adjusted}} = \frac{V - V_{\text{naïve}}}{1 - V_{\text{naïve}}} \tag{2.5}$$

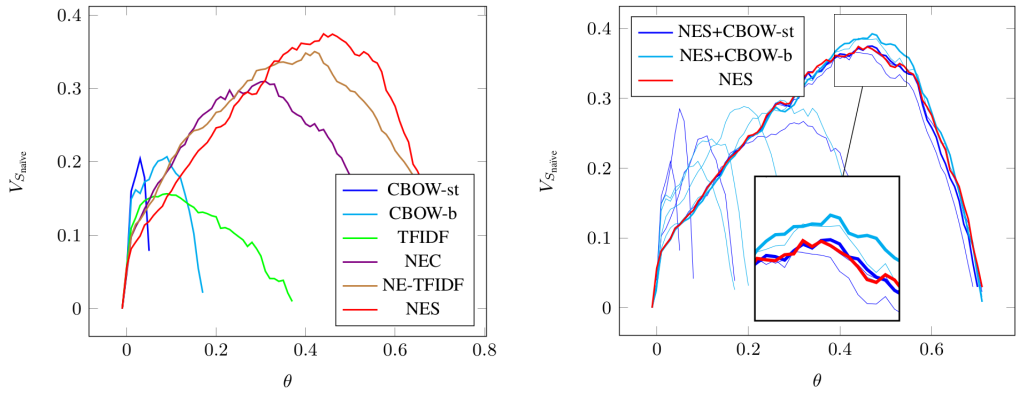
where V is a V-mesuare obtained by a certain clustering method and $V_{\text{naïve}}$ is a V-mesuare obtained by the naïve strategy, i.e., each document in its own cluster.

2.2.3 Experiments

We conducted a series of experiments to find out the best clustering method for our data. In all our experiments we used vector representations of documents and run agglomerative clustering, to produce a dendrogram with the documents as leaves. Then groups are obtained by cutting at a distance threshold θ, which is the maximum distance between any two documents in a cluster.

Our main goal was to compare various document representations to find the best features for this task. We tried the following representations:

- Word-based features:
  - TFIDF: the most standard representation, where vector coordinates corresponds to words and where values are TF-IDF scores of words within a given document;
  - CBOW-st: "standard", commonly used, word embeddings, built on Google News [60]; a document is represented as an average of word vectors;
  - CBOW-b: domain-specific word embeddings, built using the PULS business news corpus; we use the same hyper-parameters as in the "standard" embeddings: context window size is 5 words, embedding size is 300.
- NE-based features:
  - NE TF-IDF: similar to word-based but computed using only named entities;
  - NE counts: vector coordinates are NEs, values are there counts within a given document;
  - NE salience: vector coordinates are NEs, values are there salient scores.



(a) Individual features

(b) Selected combinations

Figure 2.3: Adjusted V-measure of various text representations for news clustering task. TFIDF - tf-idf computed for words; CBOW-st - standard word2vec embeddings; CBOW-b - word2vec embeddings train on business corpus; NEC - NE counts; NE-TFIDF - tf-idf computed for names; NES - salience-based representation.

In Figure 2.3a we plot the adjusted V-measure on various values of threshold θ. As can be seen from the plot in Figure 2.3a NE-based strategies outperform word-based ones. Even the worst-performing NE-based measure (raw count, NEC) is better than the word-based strategies. TF-IDF, which is the most frequently mentioned strategy in the literature, outperforms raw counts. There are arguments *for* and *against* TF-IDF. For example, it is clear that locations that pop up in the news rarely are more indicative than frequent country names. On the other hand large companies are involved in many different activities and often appear in the news, which should not affect their relevance for a particular event. The best performing measure, which is based on *salience* (NES), completely ignores overall distribution of NEs in the corpus. However, it takes into account the *position* of an entity mentioned in the text and manages to outperform both raw counts and TF-IDF.

Although NEs are crucial to determine a story they might not be enough because different stories may involve exactly the same set of NEs. In the business domain that might be a case of biggest companies, such as Google or Apple, that constantly appear in news in different narratives, which may mention the same set of names, such as locations, CEOs and the company itself. In a more general setting this can be a case of major political events that often involve the same

names, e.g., two documents that mention Donald Trump and Washington may describe completely different stories. Thus, it is worthy to combine word-based and NE-based document representations.

Figure 2.3b shows the values of V-measure obtained by combining word embeddings and NE salience. It can be seen that, in combination with salience, embeddings trained on a small domain-specific corpus (CBOW-b) outperform those trained on a much bigger general corpus (CBOW-st), although CBOW-b alone performs worse than CBOW-st. The best combination method yields an adjusted V-measure 0.47.

2.3 Grouping within news monitoring system

From the user point of view, the main role of the grouping is to minimize redundancy and to identify trending stories. But grouping may be even more crucial for a number of inner tasks not immediately visible for the user. Grouping allows to correct or hide errors made by other modules. In the user interface we show aggregated representations of document groups: we show topics and industry sectors with high *average* probability, which means that if a classifier makes a mistake in one particular document that does not affect group representation on the whole. Similarly only NEs with high *average* salience are shown to the user. Although this feature does not improve performance for groups that consist of only one or two documents, representation of the main events, which appear in dozens of documents, is more precise.

The grouping mechanism may have other advantages, that are not used in the current news monitoring system. Currently grouping is the last step that is performed after all other information is collected for every document. All classifiers work on the document level and their output is aggregated into group view. Even if some errors are found at this step, e.g., if one document is assigned with a sector label inconsistent with labels of all other group documents, we do not try to re-classify documents and just hide the incorrect label from the user. In principle, it would be possible to apply grouping as the first step and then to classify *an entire group*, since it obviously contains more useful features than any given documents in isolation. However, this approach would require some changes in the training process since, as will be described later, all data in PULS training sets are annotated document-wise.

Another interesting topic might be a *cross-document* text summarization, i.e., construction of a short summary that briefly describes a story and includes details

found in different documents. In the user interface we try to compensate, to some extent, an absence of cross-document analysis. We also find a group medoid—a document that is closest to an average of group document vectors. Documents within a group are ordered by their distance from the medoid; thus, user sees the most representative documents first. We also perform text summarization in PULS on the document level. Nevertheless, the system may benefit from more sophisticated analysis on the cross-document level.

Finally, a salience-based representation of the document has its own limitation since it ignores any relations between NEs in the text. It is worth to investigate a discourse structure of the group. One of the ways to do it might be a co-occurence graph, such as the one exemplified in Figure 2.4. The vertexes in the plot represent NEs with the node size proportional to the salience. The edges are co-occurrences of the names within the same sentence; the more often names co-occur within a group the bolder the corresponding edge. The graph presents a group, to which belongs a document shown in Figure 2.1a; comparing these two figures it is easy to see that graph representation is richer and gives additional insights into the group content. It can be seen from the plot, that not only is Facebook the most salient company for the group, it is also an entity that connects most other topics in the text.

Currently such plots are presented to the end-user in the user interface that allow analysts to gain relational information without reading group documents in details. However, these graphs might be used even more extensively within the system, for example, to collect co-occurence statistics across the entire news stream.

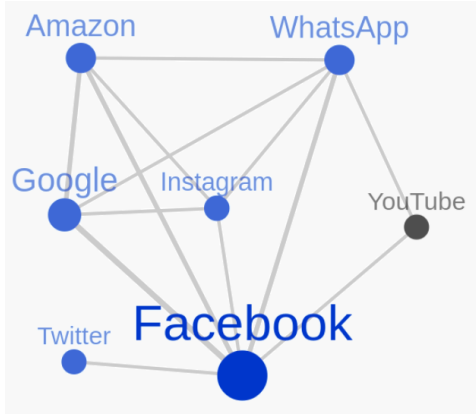


Figure 2.4: Graph representation of a group.

Chapter 3

Multi-label text classification

3.1 Problem Setting

A news monitoring system that collects thousands of news articles daily must provide the user with several ways to search and organize information. PULS allows the user to search documents using keywords and NEs but these methods may produce too narrow an output that is not necessarily pertinent to the user needs. Another way to reduce information overload is grouping but it helps only to some extent, since most groups within a given day consist of only one or two documents, which means that the system delivers thousands of news stories daily.

Thus, in addition to other search options PULS labels documents using a set of predefined categories. The main two classification schemes are *industry sectors* and *event labels*. An industry sector defines a business area, such as energy, food or engineering. An event label defines a type of event, such as investment, contract or lawsuit. To train sector and event classifiers we use a corpus of approximately 2.5 million short business reports, which were manually annotated by our partners during an earlier joint project.

The corpus has been collected over more than five years and annotated by several people. Both classifications schemes consist of several hundred labels: approximately 700 for sectors and 300 for events. Both classification are *multi-label*, i.e., a document may have more than one label. However, this option was used differently for different documents. In Table 3.1 the statistics of sector multi-labeling is presented: the first column, "Sectors per document", contains the number of different sectors assigned to a document, the second column, "Documents", contains the number of documents labeled with exactly this number of sectors. As can be inferred from the table, about 60% of the documents is labeled with exactly

Table 3.1: The sector-per-document distribution in the dataset: the majority of documents are labeled with exactly one sector.

Sectors per document	**Documents**	**Sectors per document**	**Documents**
1	1390347	13	69
2	612925	14	45
3	163736	15	23
4	44492	16	12
5	13793	17	4
6	5232	18	5
7	2219	19	8
8	976	20	2
9	560	21	1
10	328	23	4
11	173	24	2
12	125	27	1
Total number of documents:		2,235,179	

one sector; less than 5% labeled with more than 5 sectors; very few are labeled with more than 10 sectors.

It can be supposed from Table 3.1 that the our task is similar to the single-label classification problem, because the majority of the documents belong to exactly one category. However, the structure of our data is more complicated, since certain particular sectors *systematically* co-occur with others. For example, Figure 3.1 shows the number of documents labeled with *Specialist Shops* (left) or *Dairy* (right) sectors, and with other sectors. On the X axis, "1" means that the document is labeled with only *Specialist Shops* (*Dairy*) sector, while "2" or greater means that it is labeled with that total number of sectors. The Y axis represents the number of documents that are labeled with exactly this number of sectors. It can be seen from the plot that the majority of documents labeled with *Specialist Shops* are also labeled with one or more other sectors. The explanation is that documents labeled with *Specialist Shops* are frequently labeled with sectors representing a particular kind of product, such as *Jewellery* or *Sportswear*. Only 16% (3617 out of 22183, the first bar in the Figure 3.1) of documents labeled with

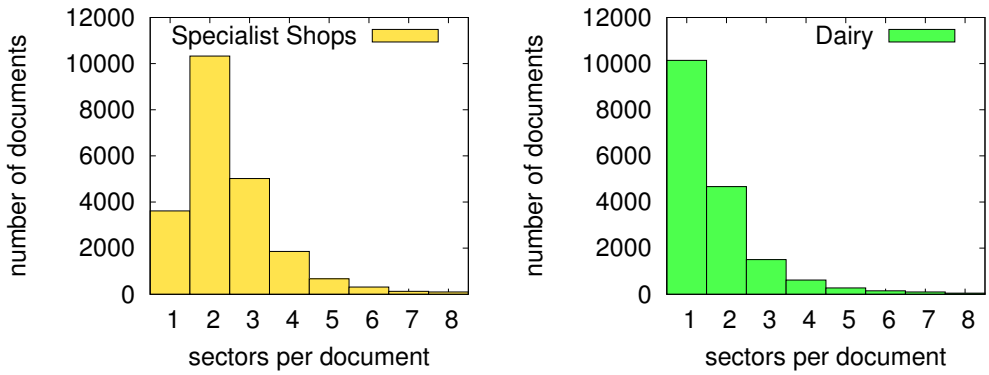
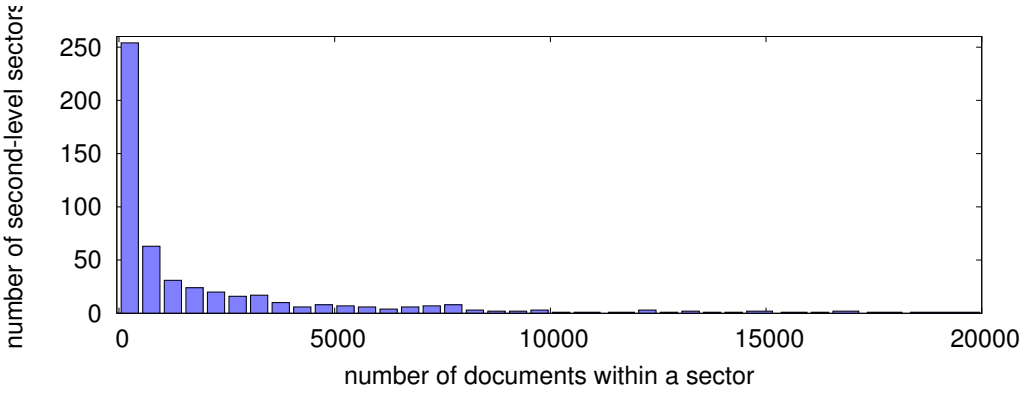


Figure 3.1: Co-occurrences of *Specialist Shops* and *Dairy* sectors with other sectors.

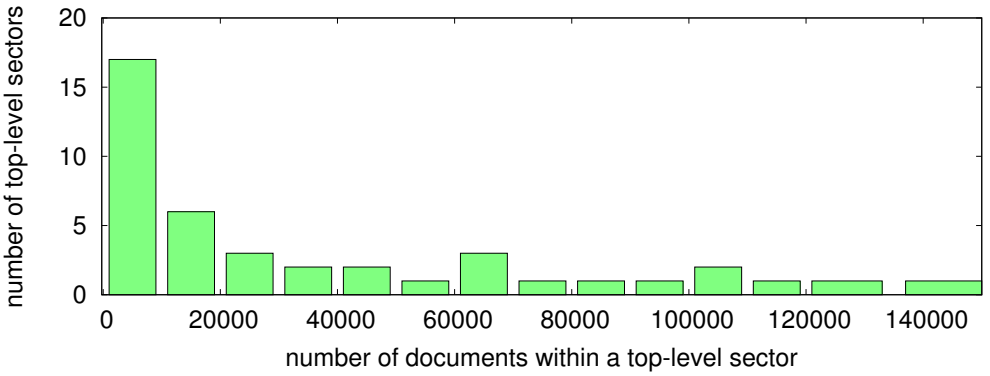
Specialist Shops are not labeled with any other category. On the other hand, 57% of documents labeled with *Dairy* are *not labeled* with any other sector. Thus, the degree of overlap is different for different sectors and we have to tackle the problem of multi-label categorization to achieve a high classification performance for the several major sectors (such as, for example, *Specialist Shops*).

The sector taxonomy, that has the biggest amount of labels, can also be organized into a two-level hierarchy. For example, *Grains, Seeds, Vegetables* and *Fruit and Berries* all can be grouped under the top-level *Agriculture*. This allows us to train classifiers at different levels of granularity; even if second-level classifiers are unable to return a particular sector, a top-level classifier still can assign a more general sector. Note that even if some sectors are considered too small to train a second-level classifier—e.g., *Plant Breeding and Research* contains only 133 positive instances—they still can contribute to the corresponding top-level sector.

As frequently happens with such detailed classification, the dataset is "naturally skewed" [58], meaning that some labels are much more frequent than others, and for some smaller labels there is almost no data to train a classifier. This happens because some events are much more rare than others, following the power law. For example, in our dataset there are only 83 documents labeled with the *Water leaks* event label, while more than 270 thousand are labeled with *New Products* because new products are launched regularly whereas water leaks do not frequently appear in business news. Similarly, 106 documents are labeled with the *Commodities: Wool* sector label while more than 100 thousand are labeled with



(a) Second-level sectors



(b) Top-level sectors

Figure 3.2: The document distribution among sector labels.

Finance: Banking. As is illustrated in Figure 3.2a, 300 sectors of 700 have less than 500 instances in our data; though a few of the largest sectors have more than 50,000 instances. Figure 3.2b demonstrates that top-level sectors are also skewed: 14 top-level sectors out of 42 have less than 10000 instances, 21—less than 20000 instances. The division of second-level sectors among top-levels is also uneven: the largest top-level sector, *Engineering* consists of 44 second-level sectors, while the smallest sectors, such as *Public Sector* or *Security* consist of 3 second-level sectors only. Note that the labels were assigned by business specialists for the use of their clients, not for training of a machine learning algorithm; they annotated data that came day by day and did not pay any efforts to balance the dataset.

Another problem we have to face is annotator bias [2], i.e., a tendency of different annotators to interpret the annotation scheme differently. This problem is exacerbated when classification is huge and hard to keep in mind. For example, a document about delivery of a new information system into the hospital may be labeled with *Information Technology: Software*, or *Health: Services and Infrastructure*, or both, depending on the annotator's preferences. Moreover, the classification scheme may change over time, as new industry sectors start playing a more important role.

Thus, even before training the classifier we had to spend significant efforts cleaning up and reorganizing the label hierarchy. During this process the minor labels were either merged or excluded. If there are several closely related labels they can be merged to produce a label with more positive instances, e.g., *Water leaks* event label can be merged with other accident events, such as *Fires* and *Hazardous Emissions*. An alternative decision might be to exclude a label from the hierarchy. This does not mean that corresponding documents are excluded from the dataset: in a multi-label setting they can have other labels and in any case they can be used as negative examples.

Similar decisions should be applied when dealing with changes in classification over time. For example, an original sector *Paints, Coatings, Adhesives* has been split into *Paints, Coatings* and *Adhesives*. We do not have any access to the inner discussion among annotators that led to this change, we have only the corpus, where some documents are labeled with one of the four labels. If we try to train a classifier using the original data it will constantly mix the more general label with more detailed ones. Thus, we should either merge all labels or discard the more general one.

Any decisions we make regrading the labeling scheme may introduce additional problems. For example, if we merge two unrelated labels together it will make training a classifier more difficult, not easier. Similarly, if a hierarchy contains errors in the grouping of the sectors into top-levels, the top-level classifier will constantly make mistakes. Thus, none of these decisions should be made based on the label names only. There are several ways to simplify the manual clean-up—building a baseline classifier and looking at the most frequently confused labels or applying unsupervised clustering on the labels [66]—but in any case, reading and comparing several documents for each label is unavoidable. Dealing with classification schemes took several weeks of work by several people, and this is a process that is rarely reflected in the literature.

3.2 Reuters corpus and comparability

As can be seen from the previous discussion, our classification corpus exhibits a number of problems that cannot be easily reproduced using commonly used text classification datasets, such as DMOZ or 20 Newsgroups. The only publicly available corpus that can be used to compare our methods with other research is RCV1 [56], a corpus of about 800000 Reuters articles published in 1996 and 1997. It has many common properties with the PULS corpus, described before: it is business-oriented, has both sector and event (topic) labels, uses a multi-label classification, which is also naturally skewed. At the same time, there are several crucial differences, since the RCV1 corpus is smaller, collected during a shorter period of time and has a cleaner classification scheme, since some effort on cleaning up the labels had been made before the corpus was published, [56]. Moreover, the RCV1 data are not suitable for experiments on fine- vs. coarse-level classifiers (top-level vs. lower-level), since the RCV1 hierarchy is not well-balanced. The labels in Reuters have the form of 5- or 7-digit codes; sectors having the same prefix are intended to be children of the same top-level sector. Others have used the first 3 digits as upper-level labels. However, these top-level labels are highly uneven: e.g., 170 distinct 3-digit codes appear in RCV1, but most of them (126) have only one child in the data. Thus classifying on the 5-digit and the 3-digit level is almost the same task.

Being a unique and valuable resource, the RCV1 corpus is frequently used in various research, though few of them are directly comparable with ours (and between each other). There are not many papers that tackle the full scale classification problem. More frequently, the authors use some subset of the data: [15, 27] use only four the most general topic labels; [23] used 6 sector categories to perform experiments with binary classification, [34] used a subset of 16 topics and 16 categories; [17] used only a subset of RCV1 with 6000 documents. Even when the entire dataset is used for experiments, the particular training-text split varies across different papers, because the original split is impractical for most algorithms: the training set was very small and consisted of 23 thousand instances only while the remaining 780 thousand were left for testing [56].

Numerical results obtained using different subsets of the same corpus are not comparable, since system performance differs, depending on the dataset. In general, the more skewed the data, the harder the classification task. For example, [18] experimented with three of the most frequently used subsets of the Reuters-21578 corpus: R(10), the set of 10 categories with the highest number of positive training examples, R(90), the set of 90 categories with at least one positive training

example and one test example, and R(115), the set of 115 categories with at least one positive training example. They demonstrate that the difficulty of the text-classification task increases as the number of categories grows. In their experiments, the mean difference in system performance between R(10) and R(90) was 25%, with respect to the macro-averaged F_1 (72% for R(10) and 47% for R(90)).

Since the RCV1 dataset is also skewed and some labels have too few instances, it would be reasonable to exclude these labels from the classification, and probably do other clean-up on the label set, such as merging labels that have the same name but different classification codes. On the other hand, since these decisions are made independently in different research groups, final results are not directly comparable among each other.

Another problem that makes it difficult to compare different studies is a huge variety of evaluation metrics that used to evaluate a classifier performance. This issue is discussed in more details in the next section.

3.3 Evaluation Measures

The most common evaluation measures are *accuracy* and *F-measure*. For a given class c, accuracy is calculated as:

$$Acc_c = \frac{TP_c + TN_c}{|c|} \tag{3.1}$$

where TP_c, TN_c, FP_c and FN_c are the number of true positive, true negative, false positive, and false negative classified instances for the class, respectively; $|c|$ is the number of documents in the test pool labeled with this class.

F-measure is calculated as a harmonic mean of *recall* and *precision*:

$$Rec_c = \frac{TP_c}{TP_c + FN_c} \qquad Prec_c = \frac{TP_c}{TP_c + FP_c} \qquad F1_c = \frac{2 \times Rec \times Prec}{Rec + Prec} \tag{3.2}$$

As can be noted from the formulae, accuracy takes into account both the true negative and true positive instances, whereas the F-measure focuses on the true positive instances only. If the data is heavily unbalanced, the accuracy measure will be less informative, since the negative instances significantly outnumber the positive instances [7, 45, 88].

To obtain an overall performance on the entire label set, the measures can be *micro-averaged* or *macro-averaged* [88]. In micro-average evaluation, first the numbers of true- and false-positives, and true- and false-negatives are counted

for all instances in the test set, and then the standard measures, e.g., recall or precision, are calculated using these numbers:

$$Rec_\mu = \frac{\Sigma_{i \in S} TP_i}{\Sigma_{i \in S}(TP_i + FN_i)} \qquad Prec_\mu = \frac{\Sigma_{i \in S} TP_i}{\Sigma_{i \in S}(TP_i + FP_i)} \tag{3.3}$$

$$\mu\text{-}F1 = \frac{2 \times Rec_\mu \times Prec_\mu}{Rec_\mu + Prec_\mu} \tag{3.4}$$

where S is the set of all classes.

In the macro-average evaluation scheme, the measures are first calculated for each class *separately*, and then these are averaged across all classes:

$$Rec_M = \frac{\Sigma_{i \in S} Rec_i}{|S|} \qquad Prec_M = \frac{\Sigma_{i \in S} Prec_i}{|S|} \qquad M\text{-}F1 = \frac{\Sigma_{i \in S} F1_c}{|S|} \tag{3.5}$$

Micro- and macro-averaging have different meaning: micro-averaging estimates a proportion of correctly classified *instances*, while macro-averaging shows how good is the classifier in average for each *class*. In the case of multi-label classification this means that for minor classes, classifiers usually perform worse than for bigger classes, [33, 78] and, as a consequence, the macro-average results are usually lower than micro-average, [18, 58, 80].

However, in many papers these measures are not reported and other ways of evaluation are proposed. For example, [57] use measures adopted from information retrieval, namely precision and cumulative gain at top K. Such works cannot be directly comparable with others, which report the F-measure, because the evaluation measures are used not only to report results but also to optimize an algorithm during training. The notion of the best classifier differs depending on which evaluation measure is used.

Thus, although RCV1 is frequently used for various experiments, we were able to find very few papers that are directly comparable with our research, in the sense that they use the entire RCV1 dataset and report micro- and/or macro-averaged F-measure.

3.4 SVM classifiers

In this section we briefly present our experiments with SVM classifiers on the RCV1 corpus, described in Papers II and III [1]. In this series of experiments

[1] We also experimented with Naive Bayes classifiers, though SVM yields higher performance.

we used a *cross-training* technique, [10]: a single *binary* classifier is trained for each label, using instances having the given label as positive examples, and all remaining instances as negative. We experimented with sector classification only, leaving topic labels aside. We mapped all seven-digit codes to their corresponding parent codes, and merge labels that have the same name but different code. After this pre-processing 245 distinct sectors remained. Since not all documents in RCV1 are labeled with sector labels, the dataset in these experiments consisted of only 350 thousand documents.

3.4.1 Dataset balancing

In Paper II we tried to find ways to treat skewness in the training data, since it imposes difficulties in building a classifier, both technical and conceptual. Technical problems arise from the fact that we train a classifier independently for each sector. If most of the documents in the training set are labeled with one particular sector that means this sector dominates in the negative instances for other classifiers. Thus, for smaller sectors we cannot be sure if a classifier is trained to distinguish its own sector or some other, major sector in negative instances. Major sectors may also dominate the training set, which leads to overfitting.

A more conceptual problem is that a classifier tends to overfit to a given label distribution that might not be optimal from a practical point of view. In most research on supervised classification it is traditional to make the assumption that not only do the test data come from the same distribution of labels as the training data, but also that the classifier will be applied in the future to data drawn from this same distribution. In reality this is rarely the case: the label distribution changes over time, even within the same news stream. For example, it is unlikely that label distribution in the current Reuters news follow the same distribution as in the RCV1 corpus published 20 years ago. Furthermore, a single set of classifiers may be required to label data from a variety of sources.

One of our goals is to build robust classifiers, which are not biased toward the particular distribution of labels in a given training set. Rather than using all available documents from a training set, we follow an *undersampling* strategy [46] and build smaller, but balanced subsets. We use a balancing procedure, suitable for the multi-label setting.

The training set and the test set are built simultaneously, starting with the sector that has the smallest number of instances. The algorithm randomly selects from the database 600 documents labeled with this smallest sector and breaks them into two classes: 3/4 of the documents for the training set and 1/4 for the

test set. If there are not enough documents, all documents labeled with the sector are used: 3/4 of them are placed into the training set and 1/4—into the test set. Then we move to the second smallest sector and count how many documents labeled with this sector are *already present* in the test and training sets—it may happen due to the sector overlap. Then the algorithm selects as many documents as it is necessary to put 450 documents in the training set and 150 documents in the test set. Then the process moves to the next smallest sector and so on. A sector can be skipped if there are more than 450 documents in the training set and 150 in the test set; or it is possible that for some sector fewer documents are used in the test or training set.

We run this process only once, which means that all the experiments described in Papers II and III have been done using exactly the same training and testing sets. After applying this procedure to RCV1 documents labeled with sector labels we obtained a balanced training set of 77.5 thousand documents. The dataset is still skewed, as can be seen in Figure 3.3, though it is much more balanced than the initial distribution.

For comparison we used an unbalanced training set, which is simply half of the corpus that includes a balanced training set. All data outside balanced and unbalanced training are used to construct development and test sets. To simulate the effect of changing trends in news streams, we generate 50 additional datasets.

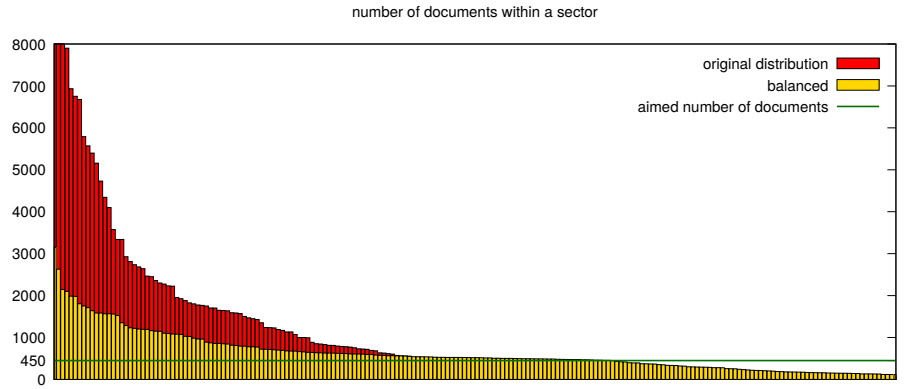


Figure 3.3: The document distribution among sectors in the original data versus the training set; 450 is the desired number of documents per sector.

To build these sets, we calculate the individual proportions of the sectors in the original distribution, then assign these proportions to 50 *random permutations* of the sector labels. We then attempt to sample 10,000 documents from the testing pool according to the new, permuted distributions. Each set among these 50 has its own label distribution, different from both the original and from each other. The distribution of labels in these random test sets will appear "naturally skewed," since it mimics the original shape. For comparison we used 10 originally distributed test set, i.e., random samples from the test pool of 10,000 documents each.

Three example test sets are shown in Figure 3.4, one "original," and two "permuted." The permuted distributions are still biased toward the largest classes in the original corpus, which is unavoidable because some larger labels have a high degree of overlap, while the smallest sectors may not have enough data to dominate the permuted distribution. However, the distributions of the permuted test sets look substantially different from the original distribution and contain significantly more instances from small- and medium-sized sectors. We use the original and permuted test sets in our comparison of balanced and unbalanced training.

The averaged results obtained on both original and permuted test sets are presented in Table 3.2. As can be seen from the table, classifiers trained on the original distribution have higher μ-F1 on originally-distributed test sets, but lower on permuted test sets; the classifiers trained on balanced training data yield higher M-F1 on *all test sets*, original and permuted.

A comparison of balanced and unbalanced training is presented in Figure 3.5, where we plot the macro- and micro-averaged F-measure obtained by classifiers trained on balanced vs. unbalanced data for each *permuted* test set. As can be seen

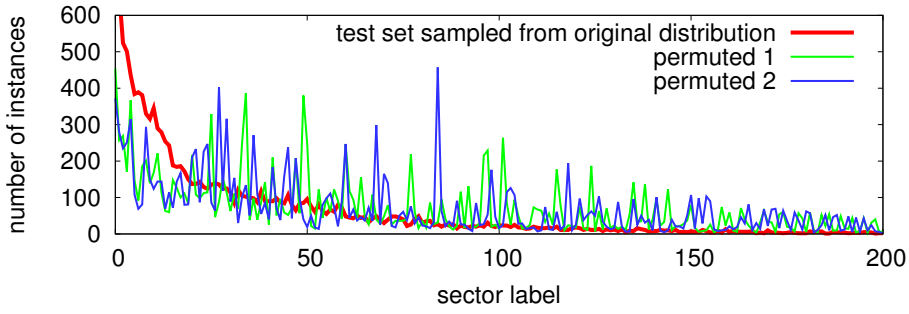


Figure 3.4: Label distributions of an original test set and permuted test sets.

Table 3.2: Results for SVM classifiers trained on balanced vs. unbalanced training sets, applied to originally-distributed and permuted test sets.

Averaged over 10 ORIGINALLY distributed test sets			
TRAINING	**Rec**	**Pre**	**F1**
	M-average		
BALANCED	**31.8**±1.3	59.1±1.1	**37.1**±1.1
UNBALANCED	24.3±0.9	**73.6**±1.3	31.8±0.9
	μ-average		
BALANCED	30.4±0.4	72.6±0.6	42.9±0.5
UNBALANCED	**36.8**±0.6	**79.5**±0.5	**50.3**±0.6

Averaged over 50 PERMUTED test sets			
TRAINING	**Rec**	**Pre**	**F1**
	M-average		
BALANCED	**32.6**±0.9	70.9±1.3	**41.8**±0.9
UNBALANCED	23.5±0.9	**74.0**±1.5	31.4±0.8
	μ-average		
BALANCED	**34.4**±0.4	**78.6**±1.4	**47.8**±0.2
UNBALANCED	29.8±1.8	76.9±1.4	43.0± *2.1*

in the top plot, the classifier trained on balanced data has significantly and consistently higher M-F1: for each test set M-F1 is over 30% higher for the balanced classifiers. As seen in the bottom plot, in the majority of cases, the classifier trained on balanced data also yields higher μ-F1 than the classifier trained on unbalanced data, although the difference between two classifiers has somewhat higher variance (also seen in Table 3.2, standard deviation scores).

Thus, the M-F1 appears to be more stable for both classifiers, which is expected, since macro-averaging gives equal weight to each class regardless of its size and thus less dependant on particular distribution. This suggests that focusing on macro-averaged results is more appropriate for real-world news classification tasks. Using training set balancing we obtained a macro-averaged F-measure higher than previously reported in the literature, as is shown in Table 3.4 and discussed in more details in Paper II.

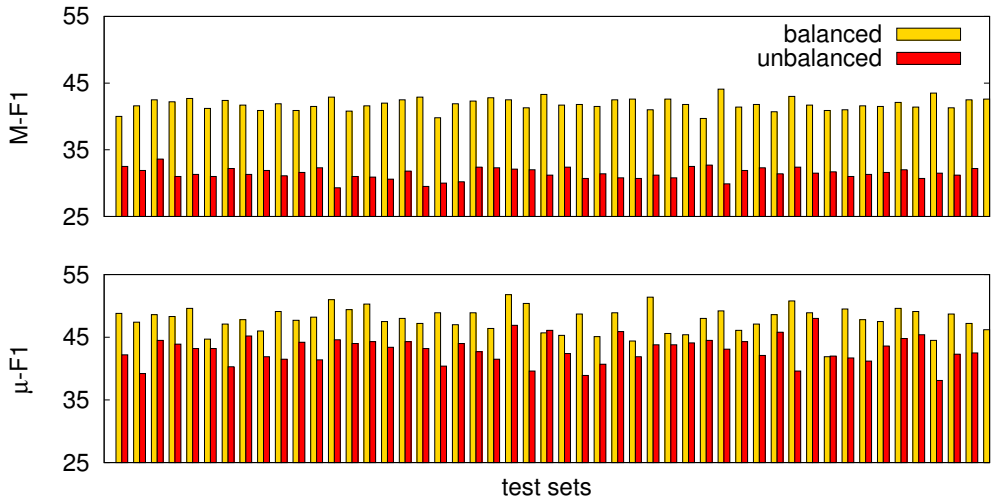


Figure 3.5: F-measures obtained by classifiers trained on balanced vs. unbalanced data, for all *permuted* test sets.

3.4.2 NEs for Text Classification

In Paper III we try to find the ways to use the PULS IE system to improve sector classification. The IE system finds mentions of companies in the corpus, using the NER module. It distinguishes company names from other proper names in the text, e.g., persons and locations. The NER module also merges variants of the same name, for example, "Apple," "Apple Inc.," "Apple Computer, Inc.," etc.

The NER module is based on a cascade of low-level patterns that find noun groups within a text. This means that the module finds not only named entities but also their *descriptors*, i.e., noun and adjective modifiers of a given name. For example, Apple can be described in the text as "computer maker" or "software giant". As can be seen in this example, a descriptor always consists of two main components: *domain*, an area in which the company works (i.e., "computer", "software") and *type*, a word that is synonymous with "company" (i.e., "maker", "giant"). A descriptor may also contain other components, such as a geographic marker (i.e., "English company") or some additional information, ("big company", "local company", etc.). A descriptor may contain all of these components, or only some of them. We use a short list of approximately 20 company words—such as "corporation", "firm", "manufacturer"—to determine the company type; we also filter out words that are too general when finding the company domain.

As a result we obtain a *profile* for each company. A profile for company Apple is presented in Figure 3.6. Among other information it contains the most frequent descriptors for the company ("IT company", "tech group", etc.). To collect these descriptors we used word lists, described above, so such descriptors as "IT company", "IT giant" and "global IT firm" are all merged together, since conceptually they contain the same information.

In the bottom of the profile we can see the most frequent industry sectors, co-occurring with the company name—top-level sectors in the upper part of the histogram, second-level in the lower. As can be seen from this illustration, each company has its own label "preferences", that is the set of industries in which it usually operates. Therefore we can find the co-occurrence of company names with industry sectors in the corpus, and use these frequencies to predict the sector labels of new documents. It is similarly possible to use company descriptors to predict sector labels; for example, we can assume that "mobile phone manufacturer" is an indicator of the *Telecommunication* sector and "dairy company" is most likely to co-occur with *Dairy Foods*.

In the middle part of the profile in Figure 3.6 we can see *related companies*, i.e., other companies that described with similar descriptors. These companies most probably operate in the same sectors. Thus, we can use information from the knowledge base collected for bigger companies to infer industry sectors for smaller companies.

As described before, a part of our corpus is hand-labeled with their true industry sectors, thus providing a link from company names to sector labels in the knowledge base. Similarly, we processed our RCV1 training set using the PULS IE system and built a separate knowledge base for our experiments. The knowledge base contains the following many-to-many relations: document-sector; document-company; company-descriptor. We tried using various combinations of these relationships to build a knowledge-based classifier. We call it "Rote" classifier, since it does not optimize any parameters during training and directly uses probabilities calculated on the corpus.

Since each document may belong to more than one sector, instead of choosing only the top-most frequent sector the classifier should return the entire sector distribution, which can be calculated using the evidence from all companies mentioned in the text. Thus, the probability that document D belongs to sector S, in the simplest case, can be defined by the formula:

$$P(S|D) = \frac{1}{|C_D|} \times \sum_{c \in C_D} P(S|c) \tag{3.6}$$

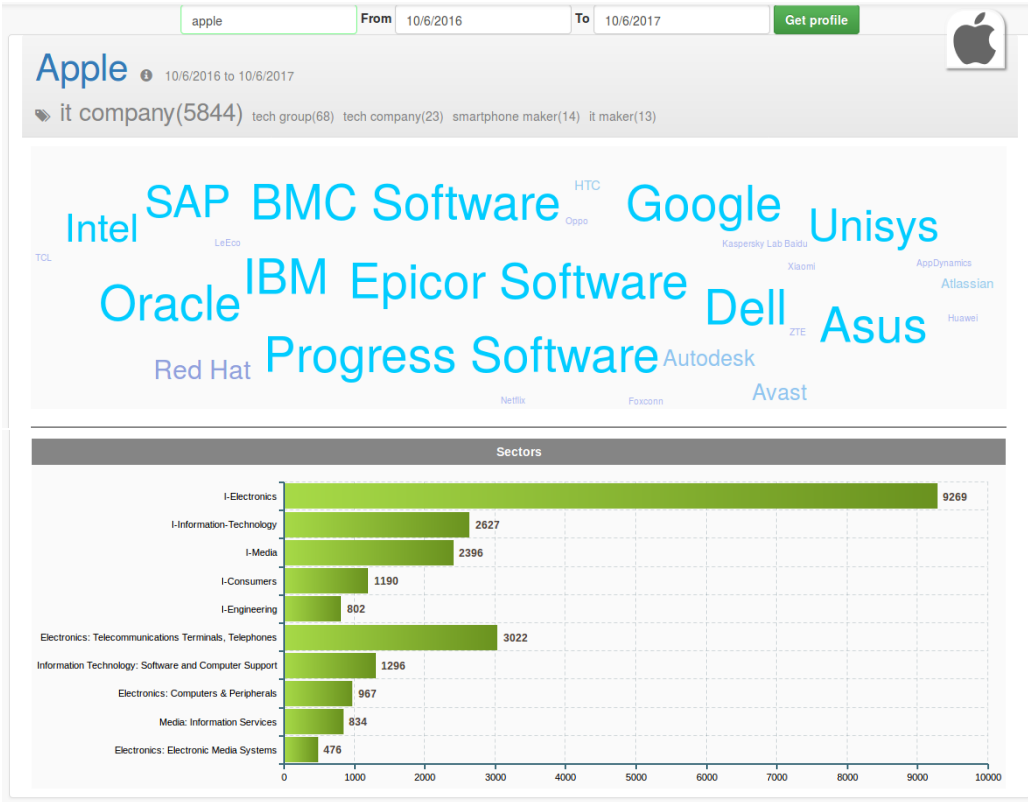


Figure 3.6: A company profile for Apple.

where C_D is the set of companies mentioned in the document, and $P(S|c)$ is the proportion of times c co-occurs with S in the knowledge base[2].

This method would be reliable if the knowledge base contains sufficient evidence to associate the company with particular sector(s). Therefore, we only use companies that appear in the corpus three or more times. This means that if a document discusses a new (or little-known) company, the name-based classifier will be unable to find a sector for the document. In this case we can use descriptors to label the document, as descriptors allow us to use evidence gained from *other* companies in the corpus.

[2]This work had been done before we worked out the current salience weighting scheme and the formula treats all document companies equally. It has been noticed by the pre-examiners that there might be more promising weighting schemes that incorporate company salience.

In this case the probability that document D belongs to sector S can be described by the formula:

$$P(S|D) = \frac{\sum_{c \in C_D} P(S|c) + \sum_{d \in d_D} P(S|d)}{|C_D| + |d_D|} \tag{3.7}$$

where d_D is the set of all descriptors mentioned in the document. Note that $|C_D| \neq |d_D|$ because in this case we can use a company descriptor even when the company does not appear in any other document in the corpus. We can also use descriptors that do not co-occur with any particular name; e.g., if the document mentions "IT companies," but does not specify company names, this mention can still be used to help classify the document.

We also tried more sophisticated ways to utilize names and descriptors, as described in Paper III, though in the experiments the method presented in Formula 3.7 showed the best results. It gave us improvement of almost 10% in both M-F1 and μ-F1 comparing with the best SVM classifier. A combination of SVM and Rote classifier allowed us to improve results from Paper II, as shown in Table 3.4.

3.5 CNN classifiers

3.5.1 Model Description

The main disadvantage of the cross-training setting described above is that each label is assigned independently. This is obviously a simplification, since the resulting model ignores many regularities in the dataset: some labels frequently co-occur, some are mutually exclusive, a document cannot have too many labels, and so on. Thus, in the next experiments we tried an algorithm that assigns all labels *simultaneously*, namely a convolutional neural network (CNN).

An overview of the CNN model is shown in Figure 3.7. The inputs are fed into the network as zero-padded text fragments of fixed size, where each word is represented as a fixed-dimensional vector (*embedding*). The inputs are fed into a layer of *convolutional filters* with multiple widths. Each convolution is a function that applied to a window of a certain size to produce a new feature:

$$c_i = f(w \cdot x_{i:i+h-1} + b) \tag{3.8}$$

where c_i is a new feature, w is a filter, i.e., a table of rational numbers of size $h \times k$, where k is the embedding size and h is an arbitrary filter size; b is a

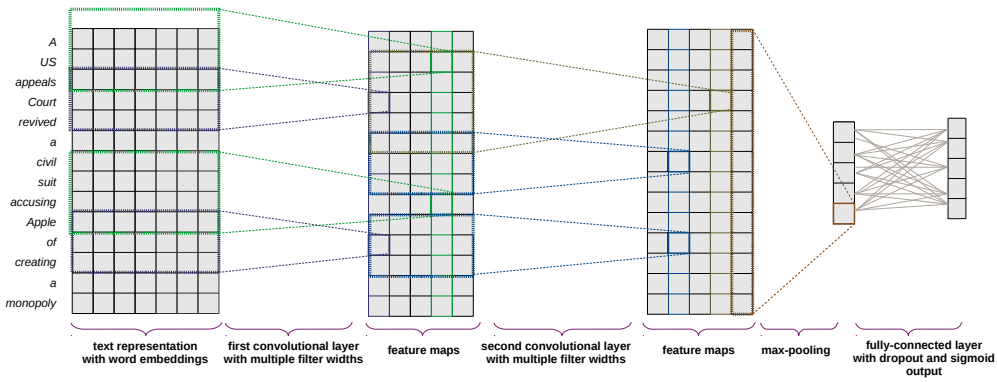


Figure 3.7: Convolution Neural Network for simultaneous sector prediction.

bias term and f is a non-linear function. A filter is successively applied to each position i using element-wise multiplication and thus produces a *feature map*. As a non-linear function we use the rectifier (ReLU):

$$f(x) = max(0, x) \tag{3.9}$$

In Figure 3.7 the embedding size is 7 and convolution of size 4×7 is schematically shown in green color. Another convolution, of size 3×7 is shown in dark blue. Feature maps, produced by each filters are then concatenated to produce a higher-level representation. This representation has the same length as the initial sentence, since convolutions are applied using padding (as is shown with the green filter in the top left corner), assuming that corresponding values are equal to zero. The width of this intermediate representation depends on the number of convolutional filters.

The idea of convolution is that the same filter is applied to all possible positions in the text. We expect that after training each filter corresponds to a certain phrase that is meaningful to determine a document label—or, to be more precise, a set of synonym phrases, since words with close meanings have similar embeddings. The first feature map can be optionally followed by deeper convolutional layers that may capture a correspondence between lower-level patterns—e.g., the fact that two phrases are close to each other in the text.

The results of the last convolutional layer are *max-pooled*, producing a vector with one scalar per filter, which contains the highest value of the corresponding

feature. In the example in Figure 3.7 a max-pooling over the last feature is shown in light brown. The max-pooled representation preserves information on whether a certain pattern is found in the text but drops information of its position. This representation is then fed into a *fully-connected layer* which produces a vector, where each dimension corresponds to a particular label. On the final step a *sigmoid* function is applied to the output:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.10}$$

Sigmoid function is a way to normalize the output and map it into [0, 1] interval, since $\lim_{x \to \infty} \sigma(x) = 1$ and $\lim_{x \to -\infty} \sigma(x) = 0$. Thus the normalized output can be interpreted as probabilities for each label.

For each label a *cross-entropy loss* is computed:

$$loss = t(l) * \log \frac{1}{p(l)} + (1 - t(l)) * \log \frac{1}{1 - p(l)} \tag{3.11}$$

where l is a label, $t(l)$ is a true value that can be 0 or 1, and $p(l)$ is a predicted probability. Individual losses are then averaged across labels and the averaged loss is used to update the weights via back-propagation.

To prevent overfitting we use *dropout regularization*, which means that at each step a random subset of parameters in the fully-connected layer remains unchanged and do not contribute to back-propagation.

This model is similar to one recently proposed for sentiment analysis [49]. The main differences are that our model uses an arbitrary number of convolutions and that we use a sigmoid function instead of softmax, since the labels are not mutually exclusive.

To train the model we use 80% of the data, 10% are used as a development set and the remaining 10% as a test set. The development set is used to determine when to stop training and to find the best thresholds for each label—if corresponding probability is higher than the threshold the label is assigned, otherwise not assigned. Finding the best threshold we optimize an F-measure for each label. Then the test set is used to obtain the final performance scores.

3.5.2 Data Representation

As is clear from the description, an input of the model is *embeddings*, continuous vector representations of the features. We train the embeddings using GloVe [65]; we used context window size 15 words and embeddings of various size (128, 300,

or 500). Features, in our case, are lower-cased *lemmas* of words. The main reason is that we use corpus-specific business embeddings, and since our corpora are relatively small, we do not have enough data to build embeddings for surface word forms. We tune the embeddings during training by updating them at each iteration.

As in our previous work in Paper III we try to take advantage of the PULS NER system. We try several options for *name representation*[3]

- **type**: each entity is represented by its type—COMPANY, PERSON, LOCATION, etc; if the type is undefined, it is represented as a general NAME; then we train the model and obtain one common vector for all company names, one vector for all person names, and so on.
- **name**: each name has its own embedding; multi-word names are interpreted as a single token.
- **name-tokens**: multi-word names are split into tokens and each token has its own embedding; the motivation for this representation is that many company names have identical parts—e.g., Air Baltic, Air France, North Star Air, Delta Air Lines—which indicates that these companies operate in the same field; modeling these name parts might be more useful than the name as a whole.
- **special-name-tokens**: similarly to the previous scheme, though tokens within names and in common contexts are distinguished; the motivation for this is that some words are used in names without any relation to the company's business area—e.g., Apple, Blackberry—and their usage in names should not be mixed with common usage.

For our experiments, we build embeddings by applying GloVe to the same corpus that is used to train the models: Reuters for experiments with the Reuters classifications, PULS for experiments with PULS. Embedding dimensionality is 128. For comparison, we also use GloVe embeddings trained on 6 billion general corpus (**glove-6B**), provided by the GloVe project[4] with embedding dimensionality of 200.

To illustrate the difference between the various word representations we present in Table 3.3 the ten nearest neighbours for example lemmas *apple* and *airline*.

[3]The name representation comparison has been published after the thesis had been submitted for pre-examination [72].

[4]https://nlp.stanford.edu/projects/glove/

When the **name** representation is used, the word *apple* is ambiguous and its nearest neighbours are both fruit words (*pear*) and computer words (*apple_computer*). In the **type** representation, the "computer" meaning disappears since all mentions of Apple as company are represented with the artificial COMPANY token. In the **glove-6B** representation, the fruit meaning is absent since all the nearest neighbours are computer-related words. Word *airline* does not have such an ambiguity and all three representations end up with more or less similar lists of nearest neighbours.

Then the **special-name-tokens** representation is used, each lemma is split in two, one for common context and one for name context. As can be seen in the table, *apple* in the common context has a clear fruit meaning and the only company that appeared in the list is juice producer Odwalla. The nearest neighbours for *apple_NE*, which is used in the name context, are mostly IT companies. However, *airline* and *airline_NE* do not have a clear distinction in meaning and the lists of their nearest neighbours are similar. In that case there is no clear advantage in using two lemmas instead of one, which would encompass both common and name usage.

Since it is difficult to estimate which naming strategy is more common—meaningful, as British Airlines, or arbitrary, as Apple—it is hard to tell in advance, which name representation is the most useful. Thus, we test all representations experimentally. The experimental results are presented in the next section. All experiments with CNN use exactly the same network structure: 3 convolution layers with filter sizes 3,7,11, 3,7,11, and 3,11, using, respectively, 512, 256 and 256 filters of each size. The models differ only in data representation.

3.6 Results and Discussion

Experimental results are presented in Tables 3.4-3.7. For the Reuters data we supplement our results with other results, found in the literature. We provide them only for reference, since experimental settings are different in different papers that makes difficult precise comparisons. For example, several previous papers use standard split, proposed when the corpus was first published, but in this split the training set consists of only 23 thousand instances and cannot be used to build word embeddings [56]. In our experiments with CNN models we use random split 80-10-10, using 80% of instances for training, 10% for development and 10% for testing. In experiments with SVM classifiers we used a balancing procedure to build the training and test set simultaneously, as described in section 3.4.1.

Table 3.3: Nearest neighbours for example words using various word representations.

apple		
name	**type**	**glove-6B**
pear	pear	iphone
unpasteurized	unpasteurized	microsoft
juice	juice	intel
apple_computer	fruit	macintosh
odwalla	salmonella	ipod
strawberry	peach	ibm
fruit	taint	ipad
macintosh	orange	software
meat	crate	google
pear_board	strawberry	itunes

airline		
name	**type**	**glove-6B**
carrier	carrier	airlines
flight	flight	airways
british_airways	passenger	lufthansa
american_airlines	aircraft	carrier
air_france	airport	flights
passenger	air	flight
lufthansa	pilot	pilots
air	route	qantas
united_airlines	plane	alitalia
aircraft	aviation	klm

special-name-tokens			
apple	*apple_NE*	*airline*	*airline_NE*
pear	computer_NE	airlines_NE	malaysian_NE
juice	macintosh_NE	airways_NE	scandinavian_NE
unpasteurized	amelio_NE	carrier	airlines_NE
odwalla_NE	operating-system	flight	system_NE
fruit	compaq_NE	air_NE	pilots_NE
anthrax	microsoft_NE	passenger	air_NE
salmonella	oracle_NE	lufthansa_NE	klm_NE
rotten	ibm_NE	pilot	passengers_NE
unpasteurised	software	aircraft	jet_NE
strawberry	jobs_NE	route	tajudin_NE

Table 3.4: Classification results on RCV1 industry sectors.

Ref.	Algorithm	M-F1	μ-F1
[56]	SVM	29.7	51.3
[92]	SVM	30.1	52.0
[75]	Naive Bayes	-	70.5
[13]	Bloom Filters	47.8	72.4
Paper II	name→SVM	56.9	63.7
Paper III	name+desc ∪ SVM	57.7	63.8
[72]	CNN type	32.2	58.4
	CNN name	61.0	80.2
	CNN name-tokens	**63.6**	**82.0**
	CNN special-name-tokens	44.3	68.3
	CNN name-tokens 6B	55.7	78.4

Table 3.5: Classification results on PULS sector labels.

		Second level		Top level	
Ref.	**Algorithm**	M-**F1**	μ-**F1**	M-**F1**	μ-**F1**
[66]	SVM + Rote	45.3	47.2	57.3	61.8
	CNN type	49.3	66.1	71.6	79.3
	CNN name	49.7	67.9	73.0	82.1
	CNN name-tokens	**60.2**	**70.3**	**78.1**	**83.9**
	CNN special-name-tokens	49.9	67.0	72.0	80.3
	CNN name-tokens 6B	48.4	67.3	72.5	81.0

In Table 3.4 we present the best results for RCV1 sector classification. As can be seen in the table, the balancing strategy described in Paper II yields macro-averaged F-measure 10% higher than in previous experiments, while using a combination of names and descriptors, as described in section 3.4.2 and Paper III, allowed us to improve this result. However, the best CNN network outperforms SVM-classifier for 6% of macro-averaged and almost 20% of micro-averaged F-measure. The possible reason is that we use an ensemble of SVM classifiers—one classifier for each label—which is unable to learn dependencies among labels.

The best data representation for CNN models for sector classification, as can be seen in Table 3.4, is **name-token**, where all tokens are treated similarly and

Table 3.6: Classification results on RCV1 topic labels.

Ref.	Algorithm	M-F1	μ-F1
[56]	SVM	61.9	81.6
[63]	ANN	69.2	85.3
[48]	CNN	67.1	85.7
[72]	CNN type	65.5	85.5
	CNN name	**66.7**	**86.2**
	CNN name-tokens	66.5	**86.2**
	CNN special-name-tokens	66.6	**86.2**
	CNN name-tokens 6B	65.8	85.8

Table 3.7: Classification results on PULS event (topic) labels.

Algorithm	M-F1	μ-F1
CNN type	22.7	48.0
CNN name	22.6	53.3
CNN name-tokens	**29.3**	**53.6**
CNN special-name-tokens	29.2	53.2
CNN name-tokens 6B	28.0	52.3

each token has exactly one embedding regardless of its usage in proper-name or common-name context. The worst performing representation is **type** where names are mapped into concepts (COMPANY, PERSON, LOCATION, etc.) and each concept has one embedding. This demonstrates the importance of named entities for sector classification and supports the assertion we made in Papers II and III that there is correlation between company names mentioned in the text and sector labels.

Note that in this experiment we use training and testing data derived from the same corpus, which means the testing and the training sets are collected from the same sources during similar periods of time and thus contain similar sets of names. In a more realistic setting a model is trained on some corpus and then applied to slightly different data, which may present different dependencies between sectors and names.

Experimental results for PULS sector classification are presented in Table 3.5. They are quite similar to what was obtained on RCV1 corpus: CNN models significantly outperform SVM classifiers; the best data representation is **name-**

tokens, the worst is **type**. The results are consistent for both second level and top level, though top-level performance is better since there are fewer top-level labels and more training data for each label. Interesting that **name-token** data representation yields much higher macro-averaged F-measure than any other method, although the difference in micro-averaged F-measure is not so great. This means that this data representation allows model to learn features important for classes with a small number of instances.

The results for RCV1 and PULS topic classification are presented in Tables 3.6 and 3.7. As can be seen in the tables, the best data representation method is again **name-tokens**, though the difference between representations is not as big as in the case of sectors, and using **type** does not lead to such a drop in model performance. This suggests that proper names are less important for event and topic classification. That corresponds to our intuition: since any company may sign a contract or be involved in a lawsuit there is little use in name features. At the same time, dependencies exist between company sectors and the most probable events: e.g., mining companies never launch new products. This is a probable reason why the **name-token** representation turned out to be the best for topic classification. The next step might be to build a model that simultaneously learns topics and sectors; we leave this for future work.

The results for PULS event (topic) classification, presented in Table 3.7, are quite poor for all five models. We address it to noise and inconsistencies in the annotation. The thorough procedure of classification clean up, which has been applied to PULS industry sectors, have not yet been used for events.

In our experiments with CNN models we did not apply any special treatment for name descriptors, assuming that since a convolution takes as an input n-grams rather then single words it should be able to learn useful descriptors automatically. Nevertheless, explicit usage of descriptors still might improve the model performance; we leave this for future work.

3.7 Error Analysis

Our classifiers can be heavily penalized for returning sectors which are relevant to the document, but were not present in the set of original labels. Since there is a large number of sector labels and documents, it may happen in some cases that correct labels were left out or extraneous labels were annotated through an annotator error. This is especially true for the PULS corpus: it has been collected over many years and during this period annotation guidelines might change. For

this reason there is uncertainty to what extent the current evaluation methods provide a fair assessment of performance.

In our experiments with PULS sector classification we conducted two types of error analysis: first, to determine the rate of sectors that are judged as "misclassified" that should be considered in fact relevant, and second, to determine the rate of original labels that should not be inferred from the text.

We first look at each false positive sector label, and the text of the document it was applied to, and decide whether this label is A. highly relevant, (meaning the sector is related to the main focus of the article), B. partly relevant (meaning some aspects related to that sector are present, but it is not the primary focus), or C. not relevant. In the example in Figure 3.8, evaluation returns precision and recall 0, we would have $pre = 50\%$, $rec = 33.3\%$ if we now assume that highly relevant sectors also are present in the original labels, and $pre = 100\%$, $rec = 50\%$ if we also include partially relevant sectors.

For 135 randomly selected test documents, classified by our best two-stage classification system (SVM-IG + Rote), we return 191 false positive labels. Our error analysis shows that of these 191, 44.0% of the labels (84 instances) could be considered highly relevant. An additional 46 false positives were partially relevant, which, when combined with the highly relevant labels, yields a rate of 68.1%.

Then, looking only at the original labels for each document, we decide whether the label is A. highly irrelevant, or B. partially irrelevant, which would occur if the label was included due to a minor mention in the article (i.e., a company name).

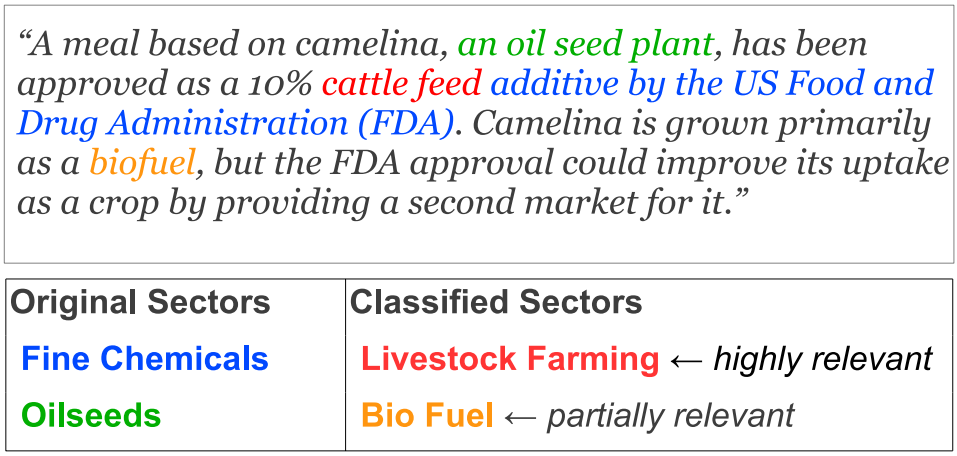


Figure 3.8: Misclassified analysis of false-positive labels.

Using another set of 170 random test documents, which contain 525 original sector labels, we found that 75 labels (14.3%) were highly irrelevant, and another 56 were partially irrelevant, for a combined rate of 25.0%.

This simple analysis points to a large disparity between the evaluation results, and what could be considered the "true" performance of our classification system.

Chapter 4

Business polarity detection

4.1 Problem Setting

In this chapter we present a work aimed at detection *polarity* of company news. Generally, polarity can be positive or negative. For example, if a company launches a new product or signs a new contract, that is viewed as a positive event; if a company recalls a product or is involved in a fraud or a bankruptcy, that is considered negative. News polarity, or "media pessimism", is a "proxy for investor sentiment or noninformational trading", i.e., investors' behavior that is not determined by knowledge of fundamental assets and rational inference [82]. Thus, research in text polarity detection in the business domain is usually inspired by practical needs of stock prediction. However, prediction of stock movements directly from text is a very difficult task, since stocks depend on many factors, including fundamental assets, general market situation and long-term history—this information is usually not presented in a single news article, which in some cases makes text and stocks completely unrelated even if they are collected on the same date. The current performance on the task is not suitable for production, as it is shown in Table 4.1, where we overview the most recent papers that use neural networks for stock prediction from text.

According to a survey of text mining for market prediction [64], the vast majority of research uses SVMs, Naive Bayes or rule-based techniques, and most of the reviewed applications rely on the manually or semi-automatically built dictionaries. The authors remark that results of most of these studies are questionable and report accuracy in the range 50-70%. Similar scepticism is expressed in another paper [37]; they argue that despite importance of automatic text analysis, companies that build their strategy fully on automatic analysis are usually not

Table 4.1: Stock prediction from textual data.

Ref	Year	Data	Method	Accuracy
[31]	2016	financial disclosures	Recursive autoencoder	56.0
[52]	2017	financial disclosures	LSTM	58.0
[21]	2015	financial news	CNN	65.5

successful. Thus, it might be more important to provide business specialists with up-to-date news analytics than try to replace them with fully automatic systems.

Our goal in Paper IV and in polarity prediction work in general is to detect a business polarity of companies mentioned in a particular article as it is expressed in the text, without any relations with stock movements or general company situation. This information is then summarized within a given group and presented to the user. In addition, company polarity is accumulated in the database, which allows users to see long-term trends, as illustrated in Figure 4.1. In our current work we leave the stock prediction task completely aside. However, we can hypothesise that stock prediction from news streams should be based on all aggregated information rather than on a single documents.

4.2 Data

4.2.1 Available datasets

We define our task as entity-level detection of business polarity for a given company within a given document. As far as we are aware there have been no datasets suitable for this particular task, though there are a few similar datasets.

A shared task on fine-grained sentiment analysis of financial microblogs and news has been organised recently as part of SemEval [14], and provided a dataset containing company names. However, this dataset contains only one thousand twits and one thousand news headlines, some of which are too short to be meaningful. We participated in the shared task [69], though this task is different from what we are trying to implement to our practical needs. In the PULS project we try to utilize the entire article, where a given company can be mentioned several times.

A corpus of 5000 business sentences has been manually annotated [59, 81], though most of its instances contain no company names, and hence cannot be used for entity-level polarity prediction.

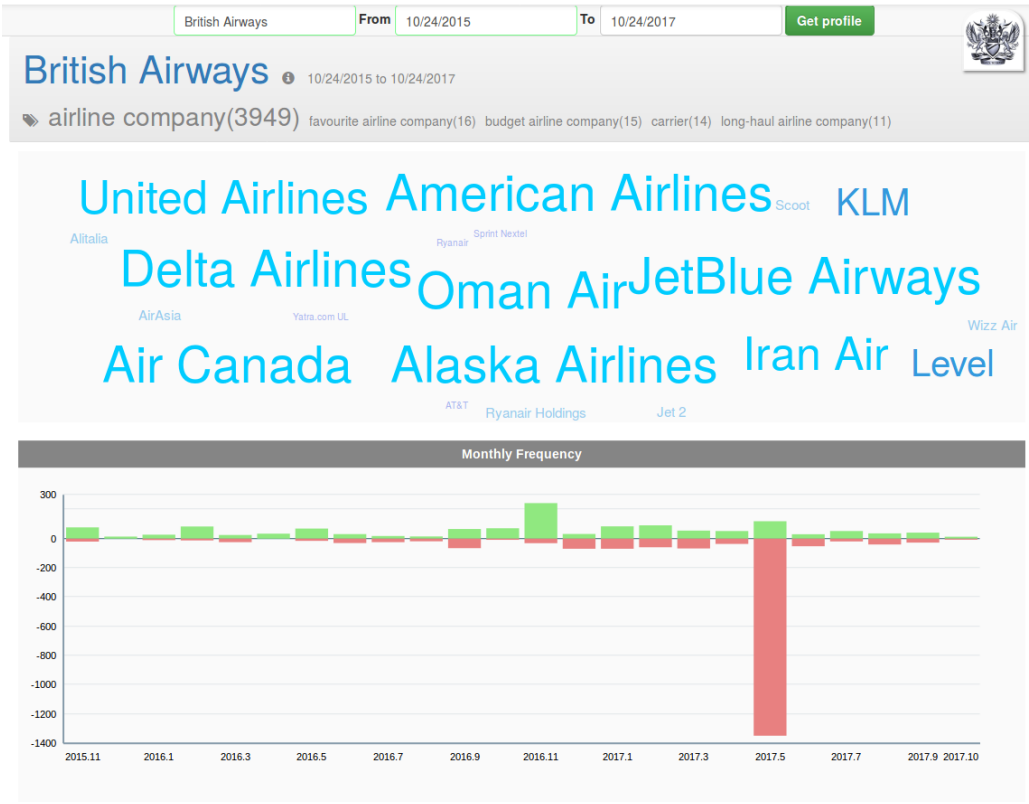


Figure 4.1: A company profile for British Airways with polarity information collected over the last two years. The spike of negative events corresponds to massive delays and cancels of company flights.

A corpus of 13000 financial disclosures has been published recently [31]. Each instance in this corpus consists of several sentences and describes a single event. The data do not have entity-level annotation, though it can be easily added, since each instance is related to one company. The main drawback of this corpus is that it was not manually annotated and market returns simultaneous to the news were used to annotate the data. In some cases this leads to inexplicable annotations. For example, appointments, that could be seen as neutral events, are annotated as either positive or negative, which cannot always be explained by the information given in the text. Thus, this dataset is aimed at stock prediction from a single document, which, as was described before, is a problematic setting.

There exists a dataset of 679 sentences in Dutch, annotated with entity-oriented business sentiment [87]. Using quantitative and qualitative analysis the authors demonstrated that a. in financial news, not all sentiment expressions within a sentence are related to the target company; b. sentiment is often expressed implicitly.

An entity-level sentiment analysis draws attention outside the business domain, e.g., a small corpus of 70 documents has been annotated with opinions, opinion targets and opinion holders [20].

4.2.2 PULS polarity dataset

A novel dataset, released together with Paper IV was manually annotated using the PULS user interface, presented in Section 1.1 and the grouping mechanism, described in Chapter 2. The task, which the dataset aimed to solve, is to determine polarity of company *mentions* within a document. A company can be mentioned several times in a document, with different polarities. In principle, we should have annotated all company names in a document. However, this would be arduous and unnecessary work, since only salient entities are shown to the user. Thus, we annotate only salient names, i.e. names mentioned at the beginning of the document. We assume that several beginning sentences tell the main piece of news and two or three first mentions of a given company have the same polarity.

The interface supports labeling polarity for companies within groups—very positive, positive, neutral, negative, very negative, and *contradictory*. The last label indicates that the document expresses contradictory opinions, or describes events with opposite polarities. Our decision to assign such examples a class of their own is similar to the annotation scheme that was proposed at the SemEval 2016 Shared Task on Aspect-Based Sentiment Analysis: contradictory opinions about the same aspect were labeled as *conflicts* [73]. For an example of contradictory polarity, consider the sentences below:
D1: *Volkswagen AG's emissions-cheating scandal took a hefty dent out of sales of its VW brand in 2016, but strong growth in China and Eastern Europe helped offset declines in other major markets.*
D2: *Samsung Electronics' profits leaped in the fourth quarter despite the humiliating Galaxy Note 7 recall that hammered the reputation of the world's largest smartphone maker.*

Documents in the groups are usually highly homogeneous, and in the majority of cases all documents in a group have the same polarity for a given entity. For trending events, a group may reach 100–200 documents; thus, using the GUI,

it is possible to annotate dozens of instances quickly by skimming only a few documents.

Still, annotating by groups may introduce some noise; e.g., the following two articles are in the same group:
D3: *Valeant Pharmaceuticals International Inc., the embattled Canadian drug-maker, agreed to sell about $2.1 billion in assets to get cash to streamline its businesses and begin easing its debt burden.*
D4: *L'Oreal to buy three skincare brands from Valeant for $1.3 billion. The French cosmetics giant paid nearly eight times the brand's combined annual revenue of $168 million.*

Here, document D4 describes the same event as D3 but gives no negative information on Valeant; this instance should be annotated as positive, even though the rest of the group is annotated as negative for Valeant. To find such examples we implemented a command-line error analysis interface. This interface presents to the user all instances ranked by *loss*, i.e., the difference between the annotated polarity and the model's prediction, and allows her to correct the labeling. High loss mean that the model not only made an error, but also that it is highly confident of its response. We expect that, if the annotation contains any serious mistakes, these instances should show up at the top of this ranking scheme. After we trained a baseline model, we applied this error analysis process iteratively to identify documents whose polarity differed from the polarity of their group. Approximately 5% of the instances were re-annotated using this procedure; the procedure also ensured that the most problematic documents were reviewed by at least two annotators.

In total, 5 people were involved in the annotation process. Though most of the documents were not problematic and were annotated by only one person, we discussed the most difficult cases until reaching consensus. In total, we annotated 1,520 groups containing 17,354 documents with 19,689 company names.

A high number of repetitions of a story is characteristic of business news; many messages are near-reprints of press-releases from larger news agencies' articles. We use the first five sentences of each document, starting from the first mention of the focus company, as training instances. If these sentences were seen before, such an instance was excluded from the training even if the documents were not identical. In our current experiments we also exclude all contradictory documents, though they stay in the dataset, along with the duplicates, and may be useful in the future.

The data distribution among the classes is shown in Table 4.2. The data were split into five folds to perform cross-validation, taking groups into account: we

Table 4.2: Class distribution in manually annotated data.

Class	# instances
very positive	2709
positive	4001
neutral	285

Class	# instances
very negative	2532
negative	4645
contradictory	146

assure that all documents in the same group must lie within the same fold, so that model is trained to distinguish polarity, not the particular types of events.

4.3 Models

To solve the polarity prediction problem we use two CNN models: a **token-based** model that uses embeddings for single words as input [49], and a **region-based** one that tries to infer task-specific embeddings for n-grams [47, 48][1]. Both models were initially proposed for various text classification tasks. Since polarity detection is an *event-level* classification models should be expanded to take into account the position(s) of the target company within the text, which we refer to as *focus*. The reason for introducing focus is that an article may mention more than one company with different polarities.

4.3.1 Token-based model

An overview of the token-based model is shown in Figure 4.2. The model is similar to the convolutional network we used for multi-label text classification, which is discussed in Section 3.5.1 and shown in Figure 3.7. The differences are in the inputs and the outputs.

The inputs are complemented with a scalar indicating the *focus*. The focus vector is shown in darker grey, with the company position framed in red. This provides an additional dimension to the word embedding, and is crucial for distinguishing between instances that differ only in focus and polarity. For experiments with polarity we use **type** vector representation, meaning that all NEs of the same type are mapped to the same token; e.g., all company names have the same embedding, person names another, etc. The reason for that is that the polarity training

[1]The token-based model is described in Paper IV; the paper describing region-based model has not yet been published.

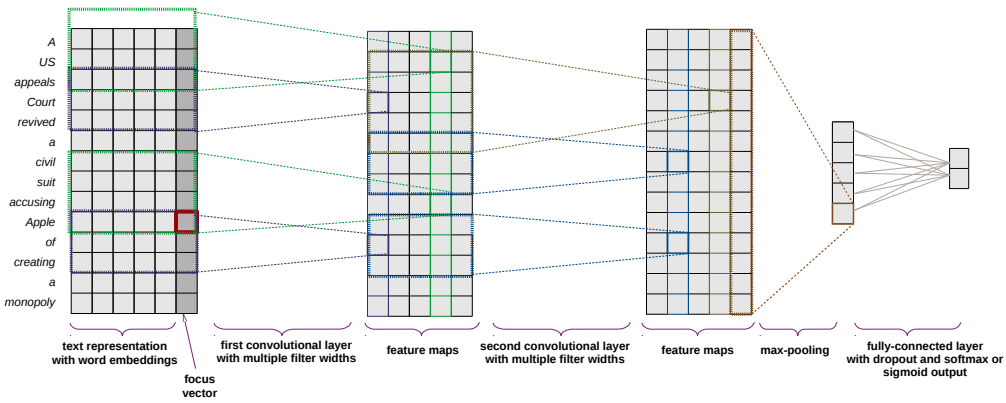


Figure 4.2: Token-based model architecture with focus vector and two convolution layers.

corpus is relatively small and the model risks to overfit to particular names that appeared in these documents by chance. We tune the embeddings during training by updating them at each iteration. This allows the model to learn word properties that are significant for sentiment detection, such as the difference between antonyms, which may not necessarily be captured well by the initial embeddings.

The output is a 2-dimensional vector that is interpreted as a probability distribution over the two possible outcomes: positive and negative. In manual annotation we use five values: "very negative" [1 0], "negative" [0.7 0.3], "neutral" [0 0], "positive" [0.3 0.7] and "very positive" [0 1]. The model may output any possible distribution. We use a fully-connected layer with dropout and softmax normalization to obtain the output distribution.

A cross-entropy loss function is computed between the network's output and the true value; the loss is used to update the weights via back-propagation.

4.3.2 Region-based model

The key component of the second model is *region embeddings*, computed according to the following formula, which defines a convolutional operation:

$$f(W \cdot r + b), \tag{4.1}$$

where r is a binary vector whose dimensions correspond to the indices of words in the vocabulary—ones indicate the words that are present in the region, and zeros in

all the other positions. These can be thought of, e.g., as "N-hot" vectors for regions of size N. W is an embedding matrix, b is a bias, and f is a rectified linear unit (ReLU). Conceptually, the regions correspond to n-grams in the text; embeddings for regions of various sizes are learned during training, which allows the model to capture task-specific expressions. At the same time, a region embeddings is a sum of token embeddings. Thus, a region representation can be meaningful even if a particular combination of words has never been encountered together during training, and using bigger region sizes does not lead to dimensionality explosion in the feature space.

An overview of the region-based model is shown in Figure 4.3. The model takes several inputs in parallel, which might differ in region size and stride; in the foreground we show regions of size 2 with stride 1. The inputs are zero-padded sentences of fixed size, represented by a *set* of word embeddings, one for each region size; in our implementation, we add a focus vector to each embedding. The inputs are split into regions and their embeddings computed by formula 4.1. We apply a local response normalization [47, 53] to the output of f.

Region embeddings are fed to a pooling layer—either max or average pooling—which is applied dimension-wise and produces an output of the fixed size, so that the representations can be concatenated for various region sizes. In this step an instance is represented by a set of vectors, where each vector represents a *segment* of the text; e.g., in the example in Figure 4.3 the text is split into three segments. The final layer is fully connected, regularised using dropout, and followed by softmax to obtain the class predictions.

4.4 Knowledge Transfer

Despite being considerably larger than any existing dataset for business polarity detection, our dataset is still small compared to what is typically used for text classification problems. Thus our main goal in Paper IV was to utilize a bigger corpus not annotated for polarity. As described in section 3.1, we have a corpus of 2.5 million short business reports, which were manually annotated using, among other information, a set of *event labels*. Some event types have clear relation to polarity: e.g., contracts or joint ventures are positive while product recalls or scandals are negative, though others, such as nominations, may happen both in positive and negative context.

We attempted two different approaches with several variations: **manual mapping** and **high-level feature transfer**.

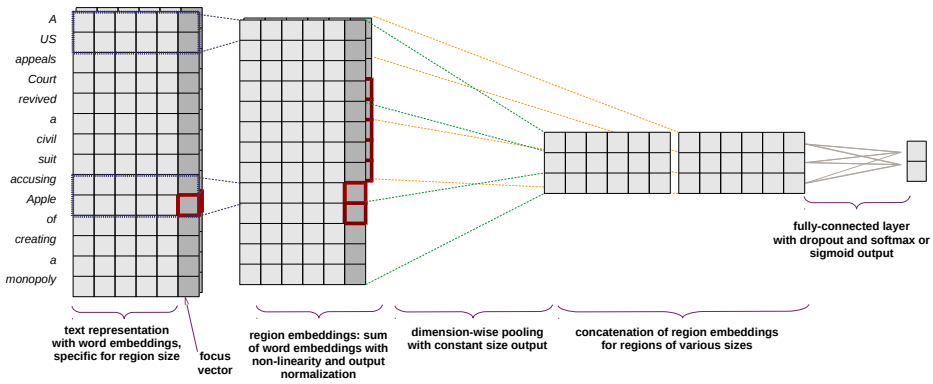


Figure 4.3: Region-based model architecture with regions of variable sizes.

4.4.1 Manual mapping

For the manual mapping, we selected those labels which we believe most clearly imply a polarity: e.g., *Investment*, *New Product* and *Sponsorship* are considered positive, while *Fraud*, *Layoff* and *Bankruptcy* are negative; in total, we have 26 "positive" and 12 "negative" labels. Using only these event labels, we constructed a training set, removing documents with labels that would result in unknown or conflicting polarities. Further, to avoid ambiguity regarding which company the label refers to, only documents whose headlines and the first sentence mention *exactly one* company were used. The reason for that is that companies that play different roles in the event may have different polarities, e.g., one company goes bankrupt and another buys its assets. We avoid these cases to keep the training data as clean as possible.

The dataset is highly skewed, with 90% of the data positive. To assure that the positive and negative training sets have approximately the same size, we apply *random undersampling* [19], i.e., we use a random subset of positive documents. Of the more than two million documents in the event corpus, 100,000 have non-ambiguous negative labels and mention exactly one company. The resulting dataset consists of 200,000 documents; 10% is used as a development set to decide when to stop training.

This newly generated *event corpus* is used in two different ways:

Tuning: a two-stage learning procedure where the model is first trained using the event corpus and then it is tunes using the small polarity corpus.

Training on combined data: in this strategy, data from both corpora are mixed together and used during training in random order.

4.4.2 High-level feature transfer

High-level feature transfer aims to reuse features learnt for a different though related task. To do so, we initially train a model to classify the event labels—using all event labels and all documents regardless of how many companies they mention. This requires a minor change in the models: since event labels are not mutually exclusive, we use a sigmoid function instead of softmax in the final layer. After the model is fully trained on the event labels, we strip off the last fully-connected layer of the network and replace it with two-class output for polarity, and then continue training using the polarity dataset. We expect that the more specific features—ones obtained closer to the output layer—will be useful in distinguishing polarity values, because of some relatedness between the two tasks. Thus, we keep almost the entire model, with the exception of the very last layer.

Since we have sufficient data labeled with events, we use 10% of the data as a development set to determine when to stop training a model (for transfer or tuning).

4.5 Experiments

Both the token-based and region-based models have a number of hyperparameters. In Tables 4.3 and 4.4 we present a number of experiments aimed at finding the best hyperparameter combintation. To compute the accuracy we interpret polarity detection as a three-way classification task: values within 0.1 of zero are considered to be neutral, all values outside it are positive or negative. Thus, the accuracy shows how often a model makes a blunder and predicts a negative polarity as positive or *vice versa*. Cosine similarity is computed using polarities mapped into continuous scale between -1 and 1[2]; this is a measure of absolute closeness between the model prediction and the manually assigned polarities, including differences between "positive" and "very positive" classes.

As can be seen from the results, cosine similarity and accuracy do not result in consistent rankings because they measure different things. For practical reasons, it is more important to us that a model does not make appalling mistakes than its ability to capture subtle differences. During manual annotation we also noticed

[2]In the result tables we present cosine similarity multiplied by 100, to enhance readability.

Table 4.3: 5-fold cross-validation Results for token-based models. Legend: **Au**—augmentation, **FP**—number of focus points per instance, **CL**—number of convolution layers, **Fi**—number of filters (of each size). Results reported in Paper IV are hightlighted with yellow.

#	Model	Focus	FP	Au	CL	Filter sizes	Fi	embeddings	dim	dropout	acc	cos
small polarity dataset												
1	baseline	-	-	-	1	3,4,5	128	glove	128	0.5	80.01	70.60
2	focus	binary	one	-	1	3,4,5	128	glove	128	0.5	79.33	69.29
3	focus	smoothed	one	-	1	3,4,5	128	glove	128	0.5	79.82	69.81
4	6conv	binary	one	+	6	3,8	40	glove	128	0.5	79.63	69.27
5	6conv	binary	many	-	6	3,8	40	glove	128	0.5	80.43	71.02
6	6conv	binary	one	-	6	3,8	40	glove	128	0.5	80.63	70.51
7	2conv	smoothed	many	-	2	3,4,5	128	random	128	0.5	78.71	69.50
8	2conv	smoothed	one	-	2	3,4,5	128	random	128	0.5	79.29	69.72
9	2conv	smoothed	one	+	2	3,4,5	128	glove	128	0.5	80.13	71.54
10	2conv	-	-	-	2	3,4,5	128	glove	128	0.5	81.22	71.51
11	2conv	smoothed	many	-	2	3,4,5	128	glove	128	0.5	**81.44**	**72.93**
combined dataset, manual mapping												
12	combined	smoothed	many	+	2	3,4,5	128	glove	128	0.5	80.98	69.24
13	combined	smoothed	many	-	2	3,4,5	128	glove	128	0.5	**82.01**	**70.92**
manual mapping + tuning												
14	tune	smoothed	many	first stage	2	3,4,5	128	glove	128	0.5	81.95	70.99
15	tune	smoothed	many	-	2	3,4,5	128	glove	128	0.5	**82.07**	**73.98**
high-level feature transfer												
16	transfer	smooth	many	-	2	3,4,5	128	glove	128	0.9	81.09	71.48
17	transfer	-	-	-	2	3,4,5	128	glove	128	0.9	81.13	72.10
18	transfer	smooth	many	-	1	3,7,11	1000	glove	300	0.5	82.22	71.25
19	transfer	-	-	-	1	3,7,11	1000	glove	300	0.5	82.47	71.46
20	transfer	binary	many	-	1	3,7,11	1000	glove	300	0.9	82.59	70.76
21	transfer	-	-	-	1	3,7,11	1000	glove	300	0.9	83.94	71.17
22	transfer	binary	many	-	1	3,7,11	1000	glove	300	0.5	82.95	**72.59**
23	transfer	smooth	many	-	1	3,7,11	1000	glove	300	0.9	**84.44**	71.76

that the distinction between "positive" and "very positive" is not always obvious for humans. Accordingly, our definition of the best model is based upon accuracy.

Focus: We experiment with three alternative representations of focus. The **baseline** model has no focus and uses only lexical features, without NEs. Technically, its focus vector consists of all zeros. This allows us to combine models: train an initial model without focus, and tune it on another input with focus. We use this feature for transfer learning.

The other two strategies are **binary** and **smoothed**. In the **binary** strategy, the focus vector contains ones in positions where the target company appears and zeros elsewhere. In the **smoothed** strategy, the focus value for each word indicates the *proximity* of the current word to the position of the nearest mention of the

Table 4.4: A selection of best-performing region-based models. Legend: **Au**—augmentation, **FP**—number of focus points per instance, **CL**—number of convolution layers, **dim**—embedding dimensionality.

#	Model	Focus	FP	Au	regions	strides	segments	pooling	embeddings	dim	dropout	acc	cos
small polarity dataset													
24	baseline	-	-	-	2,5	1,2	3	max	random	128	0.5	78.85	70.36
25	baseline	-	-	-	1,2,5	1,1,2	20	max	random	128	0.5	79.14	70.56
26	baseline	-	-	-	2,5	1,2	3	avg	random	128	0.5	79.87	71.95
27	baseline	-	-	-	2,5	1,2	20	avg	random	128	0.5	79.94	70.29
28	focus	smooth	many	-	2,5	1,2	20	avg	random	128	0.5	78.72	69.22
29	focus	binary	one	-	2,5	1,2	3	max	random	128	0.5	78.93	70.26
30	focus	binary	one	-	2,5	1,2	3	avg	random	128	0.5	79.34	71.26
31	focus	binary	many	-	2,5	1,2	20	avg	random	128	0.5	79.47	71.03
32	focus	binary	many	-	2,5	1,2	20	max	random	128	0.5	79.58	70.88
33	glove	binary	many	-	2,5	1,2	40	max	glove	128	0.5	80.30	70.33
34	glove	binary	many	-	1,2,5	1,1,2	20	max	glove	128	0.5	80.78	71.13
35	glove	binary	many	-	2,5	1,2	20	max	glove	128	0.5	81.00	72.04
36	glove	binary	many	-	2,5	1,2	20	avg	glove	128	0.5	**81.25**	**71.54**
37	huge	binary	many	-	1,2,5	1,1,2	20	avg	random	500	0.5	79.10	68.43
38	huge	binary	many	-	1,2,5	1,1,2	20	max	random	500	0.5	79.20	69.07
combined dataset, manual mapping													
39	combined	binary	many	+	2,5	1,2	20	avg	random	128	0.5	80.79	68.75
40	combined	binary	many	-	2,5	1,2	20	avg	random	128	0.5	**80.88**	**68.88**
manual mapping + tuning													
41	tune	smooth	many	-	2,5	1,2	20	avg	random	128	0.5	**79.87**	**67.88**
high-level feature transfer													
42	transfer	binary	many	-	1,2,8,20	1,1,1,2	3	avg	random	500	0.5	80.73	69.70
43	transfer	smooth	many	-	2,5	1,2	20	avg	random	128	0.9	81.18	72.12
44	transfer	-	-	-	2,5	1,2	20	avg	random	128	0.9	81.62	73.58
45	transfer	binary	many	-	1,2,8,20	1,1,1,2	20	avg	random	500	0.5	81.93	71.35
46	transfer	smooth	many	-	1,2,8,20	1,1,1,2	20	avg	random	500	0.5	82.42	72.35
47	transfer	-	-	-	1,2,8,20	1,1,1,2	20	avg	random	500	0.9	82.81	72.37
48	transfer	smooth	many	-	1,2,8,20	1,1,1,2	20	avg	random	500	0.9	83.01	73.07
49	transfer	binary	many	-	1,2,8,20	1,1,1,2	20	avg	random	500	0.9	**83.03**	**73.20**

target company. The proximity is computed according to the following formula:

$$Proximity(p) = \frac{1}{1 + |p - m|} \tag{4.2}$$

where p is the position of the current word and m is the position of the nearest mention of the target company. Thus, the proximity is 1 for a company mention, 1/2 for its immediate neighbours, 1/3 for the next neighbours, etc. It is never 0, which allows a convolution filter to use information about focus points, even if it exceeds the filter length.

We also manipulate the training instances where the same company is mentioned several times, by considering instances with (**many**) foci or splitting them into several instances with only **one** focus point.

Dropout: Dropout prevents models from overfitting by turning off a random subset of the nodes in the last layer. At each stage, we only consider the resulting

thinned net for computing the output and propagating the gradients. Doing so at the last fully connected layer, we effectively remove half of the filters or region embeddings. A dropout of 0.9 means that we consider the sub-net with 10% of the nodes in the last layer. In transfer learning that means that the model shifts its weights partially to polarity labels; the higher dropout the slower training and thus the model can make more iteration before it overfits.

Embeddings: We either use random initialization for the embeddings or Glove vectors trained on our business corpus. For the token-based model, we use Glove vectors in most of the experiments since it was reported that pretrained embeddings perform better than random ones [49]; for the sake of comparison, we performed several runs with a random initialization. For the region-based model, we use a random initialization in most of the experiments since regions themselves is a way to train task-specific embeddings [48]. However, since the region vectors are computed as a sum over token vectors (with non-linearity), pre-trained word embeddings can be used to initialise region-based models. In those cases, we use the same Glove vectors as for the token-based model, though we produce a copy of initial matrix for each region size and these matrices are then updated during training.

Data augmentation: Since the mapped training set contains only "simple" instances—i.e., they mention exactly one company—we introduce a method for *data augmentation* which hopefully generates more realistic data. By feeding our model instances that mention *several* companies, we force the network to make use of the focus information, so it can learn to handle more complex test instances, producing a better model. To augment the data, we randomly select two simple instances—which gives them a 50% chance of having different sentiments—and concatenate them. We then randomly decide which of them should receive focus. As a result, we get an instance that mentions a focus company and a *distractor* company either on the left or on the right of the focus. We expect that, by using these examples, the model would learn to ignore sentiment signals if they are far removed from the focus. We apply this procedure to our manually annotated corpus aimed to increase the number of training instances, because the manually annotated data set is relatively small.

4.6 Discussion

4.6.1 Knowledge transfer

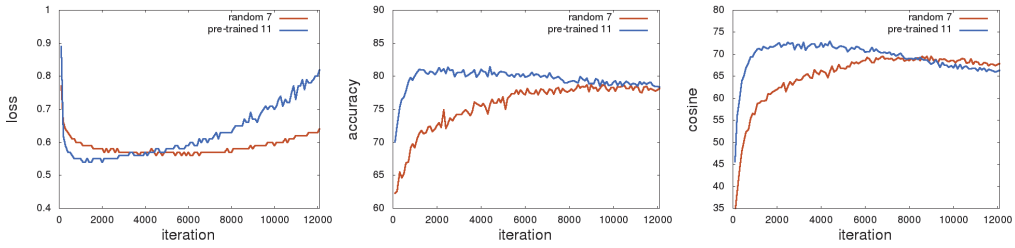
As can be seen in Tables 4.3 and 4.4, high-level feature transfer outperforms manual mapping, which is the main finding in Paper IV. The main reason might be that feature transfer can benefit from a very large corpus of 2 million documents, while only 200,000 documents can be used with the manual mapping approach, which does not allow us to train larger models due to over-fitting.

There may be other problems in the mapped dataset, resulting from how it is created. First, it contains no articles with *neutral* polarity—if an article has no positive or negative label we cannot assume it to be neutral. For example, articles labeled *Appointments* may have a negative or a positive polarity. Second, although we choose only the most "trusted" event labels for mapping to polarity, the dataset still contains some noise: e.g., a document labeled as *Contract* and assumed to be positive may discuss a canceled contract. Third, since we use only a small subset of the labels, the dataset is highly skewed and incomplete—most event types and data are not used. Most importantly, using this data a model is trained to perform a task different from our target—it learns to distinguish not positive polarity from negative, but one (sub-)set of event labels from another. We cannot assume that the model learns polarity patterns, only that polarity correlates with certain events.

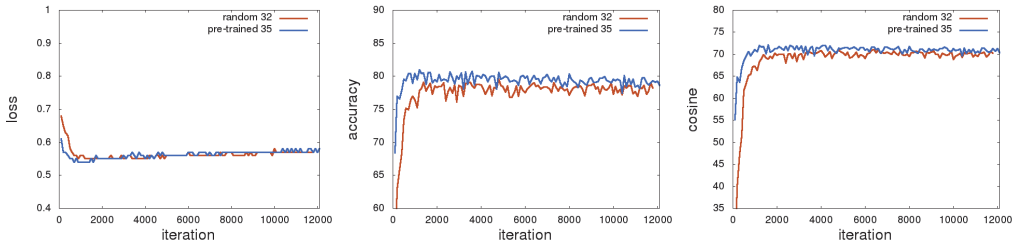
4.6.2 Embeddings

We tune the embeddings during training by updating them at each iteration. Embeddings initialization may affect not only the final performance of the model but the training process as well. In Figure 4.4a we plot training curves for token-based model 7 that uses random initialization, and token-based model 11 that uses pre-trained embeddings. All other hyperparameters are identical. As can be seen in the plots, the model with predefined embeddings learns faster—it has lower loss and higher cosine similarity and accuracy during early iterations—but overfits faster.

At the same time, a similar comparison for region-based models 32 and 35, presented in Figure 4.4b, does not show much difference in training curves, though the model with pre-trained embeddings (35) yields higher numerical values of accuracy and cosine similarity as can be seen in Table 4.4.



(a) Token-based models



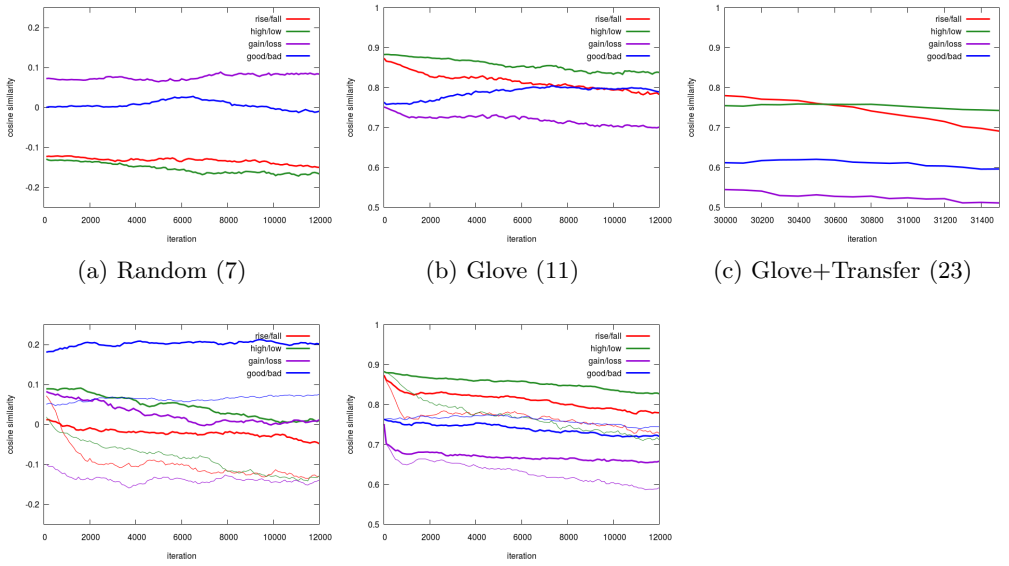
(b) Region-based models

Figure 4.4: Training curves for models that use random initialization vs models that use GloVe embeddings trained on PULS business corpus.

In our experiments embeddings tuning did not lead to the effect of antonyms resolution, which is described in the original paper, where the model was first introduced [49]. In their initial embeddings *good* was a closest neighbour for *bad* but after the training with embedding updating this was no longer the case. In Figure 4.5 we plot *cosine similarity* for several antonym pairs on various stages of training.

In Figure 4.5b we present model 11, which was trained using only polarity data. As can be seen, the semantic similarity between the words *good* and *bad* increases during training. This is a counter-intuitive result, which probably means that these words are not significant for business polarity. Other antonym pairs that consist of words with more obvious business connotations—*rise-fall*, *high-low*, *gain-loss*—behave during training in a more predictable way. Semantic similarity in these pairs steadily decrease, though it stays quite high by the moment the model overfits.

In Figure 4.5c we present model 23, which was trained for 30000 steps to predict event labels and then tuned to predict polarity; we present the tuning stage in the plot. As can be seen in the plot, after initial training semantic similarities



(a) Random (7) (b) Glove (11) (c) Glove+Transfer (23)

(d) Random (32) (e) Glove (35)

Figure 4.5: Semantic similarity for antonym pairs during training for several models. Token-based: model 7 trained on a small polarity dataset using random embedding initialization, model 11 trained using a polarity dataset and pretrained embeddings, model 23 trained using knowledge transfer and predetrained embeddings. Region-based models: model 32 trained on a small polarity dataset using random embedding initialization and model 35 trained using a polarity dataset and pretrained embeddings. Note: plots 4.5a and 4.5d use a different vertical scale than the other plots.

for antonym pairs are already lower than for the previous model, which means that distinguishing these words helps the model, to some extent, to predict event labels. This is understandable, since some event labels describe opposite events: e.g., *Plant Closures* vs. *New Capacity*. Then antonym pairs continue diverging during tuning for polarity labels, though similarity stays above 0.5 for all pairs.

In Figure 4.5a we present model 7, which was trained using polarity data and random embedding initialization. As can be seen from the plot, in this case the initial distance between antonym pairs is around zero and it stays around zero during the training.

Region-based models use a set of embedding matrices, one matrix for each region size. In Figures 4.5d and 4.5e we present semantic similarities for region-based models with random and pre-trained initializations respectively. Both models use region sizes 2,5; plots for region of size 2 presented with bold lines, of size 5—with thin lines. As can be seen in Figure 4.5e in embeddings for regions of size 5 distances between antonyms decrease faster than for regions in size 2, with the only exception of the *good-bad* pair. In random initialization, presented in Figure 4.5d, all distances between antonyms decrease except for the *good-bad* pair, though all distances stayed within a small vicinity of zero. We can conclude that embeddings in our experiments with region-based models behave similarly to embeddings in token-based models: distances between antonym pairs grow, though not significantly. Thus, features that models use to detect polarity are more complex than just positive and negative keywords.

4.6.3 Focus

As can be seen in the results, using focus not always improves a model's performance. There are instances where the baseline model outperforms models with focus—this happens when crucial patterns lie outside the filter window around the focus company, as shown in Example 1 in Table 4.5. Nevertheless, if two companies within the same text have contradictory polarities, a model without focus can assign the correct polarity value to at most one of them, as in Example 2. These examples are quite rare in our dataset; the most usual cases when two companies are involved in the same event is for them to have the same polarity, e.g., when they strike a deal. Only 6% of instances in our dataset have a paired instance that has an identical text but different focus and opposite polarity.

Another case when focus is useful is when a document contains a lot of background information, which may mention opposite polarity expressions, as in Example 3, but estimating the number of such cases is an arduous task. Since in some cases a model using focus performs worse than a model without it, there is no clear gain in this regard. However, the best-performing transfer strategy works slightly better with focus for both token-based and region-based models, as can be seen in Table 4.3 (model 23 vs. model 21) and Table 4.4 (model 49 vs. model 47).

Table 4.5: Model comparison for token-based CNN with and without focus trained using transfer approach. The company in focus is highlighted in bold.

	Example	True score	23 focus	21 no focus	Comment
1	**Valeant** to sell Dendreon unit to Sanpower for $820 million. Canada's **Valeant Pharmaceuticals International Inc.** said its affiliate will sell its Dendreon cancer business to Sanpower Group Co. Ltd. for $819.9 million, as the drugmaker continues to shed its non-core assets *to repay debt*.	-1.0	0.022	-0.322	The model without focus performs better since the company name is mentioned far away from the polarity expression and there is another name in between.
2	American Apparel files for second bankruptcy protection US retailer American Apparel (AA) has filed for its second bankruptcy protection in just over a year. The youth-focused clothes firm, hit by years of losses, will continue trading while it negotiates a potential sale of assets to Canada's **Gildan Activewear**.	1.0	0.620	-0.008	The model without focus fails: two companies with opposite polarities involved in the same event.
3	Bailed-out **Lloyds Banking Group** reports *highest annual profit for ten years*. Bottom line profits at the taxpayer-backed lender more than doubled to £4.24 billion last year, partly due to lower PPI compensation payouts. The result marks its best performance at the UK's biggest retail banking group since 2006. The government put £20.3b into the banking group, acquiring a 43 per cent stake to save it from collapse at the height of the financial crisis. This has now reduced to less than five per cent following a series of share sales and the government has indicated that it aims to shed its remaining stake this year. Announcing the results, **Lloyds** *shares jumped* 3.6 per cent and the group said its performance was "inextricably linked to the health of the UK economy, which has been more resilient than the market expected" since the referendum on EU membership.	0.4	0.179	-0.162	The model without focus fails: negative background is expressed in the text.

Chapter 5

Conclusion

This thesis presented the PULS text processing pipeline and described in detail several algorithms used for business news monitoring, namely *grouping*, *multi-label classification* and *polarity detection*. The thesis is based on four previously published papers.

In Chapter 2 we proposed a novel algorithm for news grouping, based on NE salience, which exploits a specific structure of news articles and works in an unsupervised way. We demonstrated that combination of salience with domain-specific embeddings outperforms other clustering methods on our dataset.

In Chapter 3 we investigated several approaches to large-scale multi-label text classification. First, we demonstrated that balancing of the training set allows us to build a robust classifier that shows a stable performance on the dataset taken from various class distributions. Second, we demonstrated that using automatically extracted NEs and entity descriptors in combination with keywords yields better performance than using keyword or name features alone. Third, we proposed a convolutional neural network architecture for multi-class classification that yields higher performance than previously reported methods. We demonstrated that deep learning approach outperforms an ensemble of SVM classifiers for two different datasets. We compared various ways to represent NEs for CNN classifiers and demonstrated that NEs are important features for industry sector classification, though they are less important for topic classification.

In Chapter 4 we tackled the problem of entity-level business polarity detection. We presented two novel CNNs and compared three methods of re-using data annotated for a different though remotely related task. We demonstrated that unsupervised knowledge transfer outperforms manual mapping between the two tasks.

To obtain the results presented in this thesis much effort was spent on data clean-up, reorganization and manual annotation. Throughout the thesis we tried to demonstrate the importance of data pre-processing by means of precise linguistic analysis. All methods presented in the thesis use as an input text processed by the PULS IE system that finds NEs and other low-level entities in the text. We showed that almost every task in media monitoring can gain an advantage from special treatment of NEs.

The main difference between the PULS media monitoring system and a standard IE system is that pattern-based parsing is the first not the last step of the PULS text processing pipeline. The PULS IE engine produces features for ML components that use supervised and unsupervised techniques, including advanced deep-learning models. In this thesis we overviewed several of the ML components and demonstrated how this two-stage architecture can be used to process thousands of news articles in real-time aiming at providing the end user with insights into domain events.

References

[1] Douglas E. Appelt. Introduction to information extraction. *AI Communications*, 12(3):161–172, 1999.

[2] Ron Artstein and Massimo Poesio. Bias decreases in proportion to the number of annotators. In *Proceedings of FG-MoL 2005: The 10th conference on Formal Grammar and The 9th Meeting on Mathematics of Language*, page 139, 2009.

[3] Martin Atkinson, Jenya Belayeva, Vanni Zavarella, Jakub Piskorski, Silja Huttunen, Arto Vihavainen, and Roman Yangarber. News mining for border security intelligence. In *Intelligence and Security Informatics (ISI), 2010 IEEE International Conference on*, pages 173–173. IEEE, 2010.

[4] Martin Atkinson, Mian Du, Jakub Piskorski, Hristo Tanev, Roman Yangarber, and Vanni Zavarella. Techniques for multilingual security-related event extraction from online news. In *Computational Linguistics*, pages 163–186. Springer, 2013.

[5] Martin Atkinson, Jakub Piskorski, Erik Van der Goot, and Roman Yangarber. Multilingual real-time event extraction for border security intelligence gathering. In *Counterterrorism and Open Source Intelligence*, pages 355–390. Springer, 2011.

[6] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, IJCAI'07, pages 2670–2676, 2007.

[7] Gustavo E.A.P.A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1):20–29, 2004.

[8] Branimir Boguraev and Christopher Kennedy. Salience-based content characterisation of text documents. *Advances in automatic text summarization*, pages 99–110, 1999.

[9] Jacob Boudoukh, Ronen Feldman, Shimon Kogan, and Matthew Richardson. Which news moves stock prices? A textual analysis. Technical report, National Bureau of Economic Research, 2013.

[10] Matthew R. Boutell, Jiebo Luo, Xipeng Shen, and Christopher M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[11] Nathanael Chambers and Dan Jurafsky. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 976–986. Association for Computational Linguistics, 2011.

[12] Jia Cheng, Jingyu Zhou, and Shuang Qiu. Fine-grained topic detection in news search results. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 912–917. ACM, 2012.

[13] Moustapha M. Cisse, Nicolas Usunier, Thierry Arti, and Patrick Gallinari. Robust Bloom filters for large multilabel classification tasks. In *Advances in Neural Information Processing Systems*, pages 1851–1859, 2013.

[14] Keith Cortis, André Freitas, Tobias Dauert, Manuela Huerlimann, Manel Zarrouk, Siegfried Handschuh, and Brian Davis. SemEval-2017 Task 5: Fine-grained sentiment analysis on financial microblogs and news. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 510–526, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[15] Koby Crammer, Mark Dredze, and Alex Kulesza. Multi-class confidence weighted algorithms. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP'09, pages 496–504, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[16] Hamish Cunningham. Information extraction, automatic. *Encyclopedia of language and linguistics*, pages 665–677, 2005.

[17] Zachary Alan Daniels and Dimitris N Metaxas. Addressing imbalance in multi-label classification using structured Hellinger forests. In *The Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 1826–1832, 2017.

[18] Franca Debole and Fabrizio Sebastiani. An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and technology*, 56(6):584–596, 2005.

[19] Sareewan Dendamrongvit and Miroslav Kubat. Undersampling approach for imbalanced training sets and induction from multi-label text-categorization domains. In *New Frontiers in Applied Data Mining*, pages 40–52. Springer, 2010.

[20] Lingjia Deng and Janyce Wiebe. Joint prediction for entity/event-level sentiment analysis using probabilistic soft logic models. In *Proceedings of the 2015 Empirical Methods in Natural Language Processing*, EMNLP'15, pages 179–189, 2015.

[21] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Deep learning for event-driven stock prediction. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, IJCAI'15, pages 2327–2333, 2015.

[22] Milan Dojchinovski, Dinesh Reddy, Tomás Kliegr, Tomas Vitvar, and Harald Sack. Crowdsourced corpus with entity salience annotations. In *The 10th edition of the Language Resources and Evaluation Conference*, LREC'16, 2016.

[23] Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 264–271, New York, NY, USA, 2008. ACM.

[24] Mian Du, Matthew Pierce, Lidia Pivovarova, and Roman Yangarber. Supervised classification using balanced training. In *Statistical Language and Speech Processing*, pages 147–158. Springer, 2014.

[25] Mian Du, Matthew Pierce, Lidia Pivovarova, and Roman Yangarber. Improving supervised classification using information extraction. In *International Conference on Applications of Natural Language to Information Systems*, pages 3–18. Springer, 2015.

[26] Mian Du, Lidia Pivovarova, and Roman Yangarber. Puls: natural language processing for business intelligence. In *Proceedings of the 2016 Workshop on Human Language Technology*, pages 1–8. Go to Print Publisher, 2016.

[27] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, July 2011.

[28] Jesse Dunietz and Daniel Gillick. A new entity salience task with millions of training examples. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, EACL'14, pages 205–209, 2014.

[29] Joseph E Engelberg and Christopher A Parsons. The causal impact of media in financial markets. *The Journal of Finance*, 66(1):67–97, 2011.

[30] Llorenç Escoter, Lidia Pivovarova, Mian Du, Anisia Katinskaia, Roman Yangarber, et al. Grouping business news stories based on salience of named entities. In *15th Conference of the European Chapter of the Association for Computational Linguistics Proceedings of Conference, Volume 1: Long Papers*, EACL'17, 2017.

[31] Stefan Feuerriegel and Ralph Fehrer. Improving decision analytics with deep learning: the case of financial disclosures. In *The 2016 European Conference on Information Systems*, ECIS'16, 2016.

[32] Ingrid E. Fisher, Margaret R. Garnsey, and Mark E. Hughes. Natural language processing in accounting, auditing and finance: a synthesis of the literature with a roadmap for future research. *Intelligent Systems in Accounting, Finance and Management*, 23(3):157–214, 2016.

[33] George Forman. A pitfall and solution in multi-class feature selection for text classification. In *Proceedings of the twenty-first international conference on machine learning*, ICML '04, pages 38–45, New York, NY, USA, 2004. ACM.

[34] Evgeniy Gabrilovich and Shaul Markovitch. Feature generation for text categorization using world knowledge. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, volume 5 of *IJCAI'05*, pages 1048–1053, 2005.

[35] Mahak Gambhir and Vishal Gupta. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66, 2017.

[36] Michael Gamon, Tae Yano, Xinying Song, Johnson Apacible, and Patrick Pantel. Identifying salient entities in web pages. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2375–2380. ACM, 2013.

[37] Paul A. Griffin and Mohammedi Padaria. Is financial analysis doomed? The birth of "reactive valuation" analysis. *Accounting and Finance Research*, 6(3):39, 2017.

[38] Ralph Grishman, Silja Huttunen, and Roman Yangarber. Real-time event extraction for infectious disease outbreaks. In *Proceedings of the second international conference on Human Language Technology Research*, pages 366–369. Morgan Kaufmann Publishers Inc., 2002.

[39] Ralph Grishman, Silja Huttunen, and Roman Yangarber. Information extraction for enhanced access to disease outbreak reports. *Journal of biomedical informatics*, 35(4):236–246, 2003.

[40] David Hartley, Noele Nelson, Ronald Walters, Ray Arthur, Roman Yangarber, Larry Madoff, Jens Linge, Abla Mawudeku, Nigel Collier, John Brownstein, et al. The landscape of international event-based biosurveillance. *Emerging Health Threats Journal*, 3(1):7096, 2010.

[41] Silja Huttunen, Arto Vihavainen, Mian Du, and Roman Yangarber. Predicting relevance of event extraction for the end user. In Thierry Poibeau, Horacio Saggion, Jakub Piskorski, and Roman Yangarber, editors, *Multisource, Multilingual Information Extraction and Summarization*, Theory and Applications of Natural Language Processing, pages 163–176. Springer Berlin, 2013.

[42] Silja Huttunen, Arto Vihavainen, Peter von Etter, and Roman Yangarber. Relevance prediction in information extraction using discourse and lexical features. In *Proceedings of NoDaLiDa: the* 18^{th} *Nordic Conference on Computational Linguistics*, Riga, Latvia, 2011.

[43] Silja Huttunen, Roman Yangarber, and Ralph Grishman. Complexity of event structure in IE scenarios. In *Proceedings of the* 19^{th} *International Conference on Computational Linguistics*, COLING 2002, Taipei, 2002.

[44] Silja Huttunen, Roman Yangarber, and Ralph Grishman. Diversity of scenarios in information extraction. In *Proceedings of the Third International*

Conference on Language Resources and Evaluation, LREC'02, Las Palmas de Gran Canaria, Spain, 2002.

[45] M. Ikonomakis, S. Kotsiantis, and V. Tampakas. Text classification using machine learning techniques. *WSEAS Transactions on Computers*, 4(8):966–974, 2005.

[46] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[47] Rie Johnson and Tong Zhang. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '15, pages 103–112, 2015.

[48] Rie Johnson and Tong Zhang. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems*, pages 919–927, 2015.

[49] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP'14, 2014.

[50] Keisuke Kiritoshi and M. A. Qiang. Named entity oriented difference analysis of news articles and its application. *IEICE transactions on Information and Systems*, 99(4):906–917, 2016.

[51] Jan Kleinnijenhuis, Friederike Schultz, Dirk Oegema, and Wouter Van Atteveldt. Financial news and market panics in the age of high-frequency sentiment trading algorithms. *Journalism*, 14(2):271–291, 2013.

[52] Mathias Kraus and Stefan Feuerriegel. Decision support from financial disclosures with deep neural networks and transfer learning. *Decision Support Systems*, 2017.

[53] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[54] B Shravan Kumar and Vadlamani Ravi. A survey of the applications of text mining in financial domain. *Knowledge-Based Systems*, 114:128–147, 2016.

[55] Gaël Lejeune, Antoine Doucet, Roman Yangarber, and Nadine Lucas. Filtering news for epidemic surveillance: towards processing more languages with fewer resources. In *4th International worshop on cross-lingual information access CLIA 2010*, pages 8–pages, 2010.

[56] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. RCV1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397, 2004.

[57] Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124. ACM, 2017.

[58] Ying Liu, Han Tong Loh, and Aixin Sun. Imbalanced text classification: a term weighting approach. *Expert Systems with Applications*, 36(1):690–701, 2009.

[59] Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796, 2014.

[60] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[61] Tom Mitchell and E Fredkin. Never ending language learning. In *2014 IEEE International Conference on Big Data*, pages 1–1. IEEE, 2014.

[62] Soto Montalvo, Víctor Fresno, and Raquel Martínez. Nesm: a named entity based proximity measure for multilingual news clustering. *Procesamiento del lenguaje natural*, 48:81–88, 2012.

[63] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. Large-scale multi-label text classification—revisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 437–452. Springer, 2014.

[64] Arman Khadjeh Nassirtoussi, Saeed Aghabozorgi, Teh Ying Wah, and David Chek Ling Ngo. Text mining for market prediction: A systematic review. *Expert Systems with Applications*, 41(16):7653–7670, 2014.

[65] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP'14, pages 1532–1543, 2014.

[66] Matthew Pierce. Large-scale multi-label text classification for an online news monitoring system. Master's thesis, University of Helsinki, 2015.

[67] Jakub Piskorski, Martin Atkinson, Jenya Belyaeva, Vanni Zavarella, Silja Huttunen, and Roman Yangarber. Real-time text mining in multilingual news for the creation of a pre-frontier intelligence picture. In *ACM SIGKDD Workshop on Intelligence and Security Informatics*, page 4. ACM, 2010.

[68] Jakub Piskorski and Roman Yangarber. Information extraction: past, present and future. In *Multi-source, multilingual information extraction and summarization*, pages 23–49. Springer, 2013.

[69] Lidia Pivovarova, Llorenç Escoter, Arto Klami, and Roman Yangarber. HCS at SemEval-2017 Task 5: Polarity detection in business news using convolutional neural networks. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 842–846, 2017.

[70] Lidia Pivovarova, Silja Huttunen, and Roman Yangarber. Event representation across genre. In *Proceedins of the* 1^{st} *Workshop on Events: Definition, Detection, Coreference, and Representation*, NAACL/HLT, 2013.

[71] Lidia Pivovarova, Arto Klami, and Roman Yangarber. Benchmarks and models for entity-oriented polarity detection. In *Proceedings of The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL/HLT, 2018.

[72] Lidia Pivovarova and Roman Yangarber. Comparison of representations of named entities for multi-label document classification with convolutional neural networks. In *Proceedings of The Third Workshop on Representation Learning for NLP*. Association for Computational Linguistics, 2018.

[73] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphee De Clercq, Veronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Núria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. SemEval-2016 Task 5:

Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California, June 2016. Association for Computational Linguistics.

[74] Marco Ponza, Paolo Ferragina, and Francesco Piccinno. Document aboutness via sophisticated syntactic and semantic features. In *International Conference on Applications of Natural Language to Information Systems*, pages 441–453. Springer, 2017.

[75] Antti Puurula. Scalable text classification with sparse generative modeling. In Patricia Anthony, Mitsuru Ishizuka, and Dickson Lukose, editors, *PRICAI 2012: Trends in Artificial Intelligence*, volume 7458 of *Lecture Notes in Computer Science*, pages 458–469. Springer Berlin Heidelberg, 2012.

[76] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Conference on Empirical Methods in Natural Language Processing, Conference on Computational Natural Language Learning, Joint Meeting following ACL 2007*, EMNLP-CoNLL'07, pages 410–420, 2007.

[77] Evan Sandhaus. The New York Times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752, 2008.

[78] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM computing surveys*, 34(1):1–47, 2002.

[79] Arisa Shollo and Karlheinz Kautz. Towards an understanding of business intelligence. In *Australasian Conference on Information Systems*, volume 21, 2010.

[80] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009.

[81] Pyry Takala, Pekka Malo, Ankur Sinha, and Oskar Ahlgren. Gold-standard for topic-specific sentiment analysis of economic texts. In *Ninth International Conference on Language Resources and Evaluation*, LREC'14, pages 2152–2157, 2014.

[82] Paul C. Tetlock. Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, 62(3):1139–1168, 2007.

[83] Michal Tkáč and Robert Verner. Artificial neural networks in business: Two decades of research. *Applied Soft Computing*, 38:788–804, 2016.

[84] Hiroyuki Toda and Ryoji Kataoka. A search result clustering method using informatively named entities. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*, pages 81–86. ACM, 2005.

[85] Salvatore Trani, Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. Sel: A unified algorithm for entity linking and saliency detection. In *Proceedings of the 2016 ACM Symposium on Document Engineering*, pages 85–94. ACM, 2016.

[86] Arjen Van Dalen, Claes de Vreese, and Erik Albæk. Economic news through the magnifying glass: How the media cover economic boom and bust. *Journalism Studies*, 18(7):890–909, 2017.

[87] Marjan Van de Kauter, Diane Breesch, and Véronique Hoste. Fine-grained analysis of explicit and implicit sentiment in financial news articles. *Expert Systems with Applications*, 42(11):4999–5010, 2015.

[88] Yiming Yang. An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1):69–90, 1999.

[89] Roman Yangarber. Counter-training in discovery of semantic patterns. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, ACL'03, pages 343–350. Association for Computational Linguistics, 2003.

[90] Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. Automatic acquisition of domain knowledge for information extraction. In *Proceedings of the 18th conference on Computational linguistics*, ACL'00, pages 940–946. Association for Computational Linguistics, 2000.

[91] Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. Unsupervised discovery of scenario-level patterns for information extraction. In *Proceedings of the sixth conference on Applied natural language processing*, pages 282–289. Association for Computational Linguistics, 2000.

[92] Dong Zhuang, Benyu Zhang, Qiang Yang, Jun Yan, Zheng Chen, and Ying Chen. Efficient text classification by weighted proximal SVM. In *Fifth IEEE International Conference on Data Mining*, 2005.