Universitat Autònoma de Barcelona

TESI DOCTORAL

Linguistic Support for Protest Event Data Collection

Autor(a):
Vera Danilova, M. Sc.

Dirigida per:
Dr. Xavier Blanco Escoda,
Dr. Mikhail Alexandrov

Doctorat en Llengües i Cultures Romàniques

Department de Filologia Francesa i Romànica
Facultat de Filosofia i Lletres

September, 2015

UNIVERSITAT AUTÒNOMA DE BARCELONA

Abstract

Facultat de Filosofia i Lletres
Department de Filologia Francesa i Romànica

Linguistic Support for Protest Event Data Collection

Vera Danilova, M. Sc.

The development of approaches to structuring chaotic information streams coming from the web led to the emergence of a new powerful instrument for the study of collective human behaviour: Internet Sociology. The variety of user-generated data, both quantitative and qualitative, allowed for a research from a different perspective, without the need of traditional questionnaires. Sociologists benefit from the data coming from news, blogs, social networks, comments, etc. to get an insight into the relation between online and offline collective human behaviour trends.

Social science has a long history of research on the protest activity. Today, the systematic quantitative study of this phenomenon acquires importance in the light of the world-famous events, such as Maidan, Gezi Park protests, "Charlie Hebdo" and others. Protests are being reported almost daily by the news and are constantly reverberating across the social media. Today's news are global and discuss similar events in different languages producing lots of duplicate messages and, sometimes, contradictory descriptions. Reports are created by either individual authors or agencies and vary in their textual representation. Despite the fact that news media credibility is questionable, it is still the most used data source in the context of protest events collection.

Earlier social protest studies have been based on the manual analysis of newspapers data. In the recent years, automatic approaches have been applied to the protest database population and coding. The main drawback of the automatically populated databases is the quality of protest event coding, the produced unit of analysis. As Jay Ulfelder, an american political scientist, wrote in his blog in June of 2014, discussing the attempt of Kalev Leetaru (the creator of GDELT event database) to analyze world's protest activity intensity in the last 35 years, the ideal database would include the number of protest events, the number of participants and the deeds of those participants, however, there is no such depository for all relevant events and it is unlikely to be created.

The existing databases are collected automatically on the basis of English-language newspapers or automatic translation (in case of GDELT, from around 100 languages using Google Translate), the quality of which is still far from perfect. In his blog, Anthony A. Boyles, a computational social scientist, describes the problems of GDELT [7]. His conclusion is that GDELT is useful for measuring topics' reporting coverage. The domain of protest event data collection needs the application of interdisciplinary techniques to perform multilingual event extraction and deliver good-quality and reliable event data automatically.

This thesis addresses the problem of automatic protest event coding quality and proposes the tools for multilingual event extraction to improve the quality of analysis unit. In the course of the present work, we investigated the state of the art in protest event data collection and coding and explored multilingual event extraction systems

and techniques they use (the publicly available data). In the absence of a multilingual training dataset for supervised learning we adopt a rule-based approach to multilingual event extraction and propose a domain concept hierarchy, as well as a set of generic informational patterns and gazetteers within a GATE 8.0 pipeline for the automatic population of the gazetteers.

The present work contributes to the automatic protest event data collection and coding by the following: i) a formalized description of the protest event concept on the basis of news headlines (automatically populated concept hierarchy), ii) generic patterns and gazetteers for text processing in six languages (Bulgarian, French, Polish, Russian, Spanish, Swedish), which helps to deal with the absence of a multilingual training set, iii) multilingual corpus of texts related to protest events. The obtained data can be applied among others for the monitoring and analysis of event-specific social networks' response.

Acknowledgements

I thank my supervisors at the AUB, Dr. Xavier Blanco Escoda and Dr. Mikhail Alexandrov, and my colleagues at the RANEPA for their encouragement and valuable suggestions. I also want to thank all of my friends, including the recently qualified doctors Anton Tyukov and Yaroslav Nayden, for their cooperation, advice and critical reasoning. Above all, I express an immense gratitude to my parents and my husband Maxim, who have always supported my undertakings, and to our little son, our brightest star, our daily inspiration!

Contents

List of Figures

List of Tables

Abbreviations

ACE	**A**utomated **C**ontent **E**xtraction program
CAMEO	**C**onflict **a**nd **M**ediation **E**vent **O**bservations ontology
DoCA	**D**ynamics **o**f **C**ollective **A**ction dataset
EAT	**A**utomated **E**vent **A**nnotation **T**ool
El:DIABLO	**E**vent/**L**ocation: **D**ataset **I**n **A** **B**ox, **L**inux-**O**ption
EMM	**E**uropean **M**edia **M**onitor
EOI	**E**vent **O**f **I**nterest
GATE	**G**eneral **A**rchitecture for **T**ext **E**ngineering
GDELT	**G**lobal **D**atabase of **E**vents, **L**anguage and **T**one
HUMINT	**Hum**an **Int**elligence reports
IE	**I**nformation **E**xtraction
JAPE	**J**ava **A**nnotation **P**atterns **E**ngine
KEDS	**K**ansas **E**vent **D**ata **S**ystem
MT	**M**achine **T**ranslation
NER	**N**amed **E**ntity **R**ecognition
OSC	The U.S. Governmental **O**pen **S**ource **C**enter
PEA	**P**rotest **E**vent **A**nalysis
PR	**P**rocessing **R**esource
PETRARCH	**P**ython **E**ngine for **T**ext **R**esolution **a**nd **R**elated **C**oding **H**ierarchy
Serif	**S**tatistical **E**ntity and **R**elation **I**nformation **F**inder
SIGINT	**Sig**nals **Int**elligence reports
SPEED	**S**ocial, **P**olitical, and **E**conomic **E**vent **D**atabase
SSP	**S**ocietal **S**tability **P**rotocol
TABARI	**T**ext **A**nalysis **B**y **A**ugmented **R**eplacement **I**nstructions
TF-IDF	**T**erm **F**requency - **I**nverted **D**ocument **F**requency

WEIS	**W**orld **E**vent/**I**nteraction **S**urvey ontology
W-ICEWS	**W**orld-Wide **I**ntegrated **C**risis **E**arly **W**arning **S**ystem

To my family

Chapter 1

Introduction

1.1 Protest Event Data Collection from News Media

The Internet has evolved from a platform for publishing to a large communication system, where the circulating information is being constantly transformed, and each fact is overgrown with thousands of opinions. Currently, sociology deals with a new research object, a virtual society. The newly emerged *Internet Sociology* discipline studies sociological phenomena using data coming from the web.

News, one of the main information sources for sociologists, have also experienced changes from industrial to informational society. Earlier newspapers were limited to the publication of local news in local languages. Today's news are virtual, globally accessed, and pursue multiple (not only informational) goals. There is an enormous amount of news channels across the globe varying in focus and scale. Aggregators gather English or translated versions of reports from the whole world. Due to the existence of these multiple news sources reporting on the world politics, financial issues, natural disasters, society, etc., we face an uncontrolled overproduction of messages on the same events and reinterpretation of the same facts. Intended reinterpretation and selective coverage of events constitute the news bias problem and powerful means of public opinion manipulation. We are likely to receive a biased view of events, because we tend to "consume" our daily portions of news reports from national TV or the web in our native languages or English from time to time, and we are unable to compare, because our reading capacities are obviously limited.

Society is highly influenced by mass media that shapes the view of events and is often controlled by governing authorities, which is why news articles have always been an important source of information for social scientists. Up to the recent years, researchers

have been primarily relying on the manual collection and analysis of data. This thesis deals with integrating interdisciplinary knowledge into the automatic processing and data extraction from news sources. We introduce a natural language-based application EuroPEA for the annotation and extraction of multilingual protest event data. Multilingual natural language patterns compiled into finite state transducers constitute the core of the application.

Protest activity reflects the level of people's satisfaction, confidence and belief in leader. It has been a hot issue in the recent years. *"Presidents, prime ministers and assorted rulers, consider that you have been warned: A massive protest can start at any time, seemingly over any issue, and can grow to a size and intensity no one expected. Your country's image, your own prestige, could risk unraveling as you face the wrath of the people"* - wrote Frida Ghitis, special to CNN, in June 2013 refering to the dramatic events that took place in Turkey and Brazil. November of the same year marked the beginning of the famous civil unrest events in Ukraine, which lead to the coup d'etat and grew into an international conflict and a civil war. Collective human behaviour is the power, which is of prime interest to governmental workers and scientists. Social science has been studying the trends in the occurrence of such events by analyzing news articles published during several decades and collecting huge event databases. The publicly available event datasets that can be used for training of an automatic protest extractor are based on English-language newspapers. Obtaining data from non-English sources, news and social networks, allows a social scientist to create a more complete and objective view of happenings that already have repercussion in the whole world, as well as of the local ones that set the stage for bigger events. Manual event data collection from non-English newspapers uses human translation, which still beats the automatic one, however, it is too costly when dealing with big amounts of information. Statistical translation software neither resolves the problem, because it needs domain-specific parallel corpora for each pair of languages to avoid homonymy-related errors. An algorithm that incorporates multilingual event extraction techniques is needed to create reliable protest event data in non-English languages and improve the current state of affairs in the domain. Specifically, within the protest event data analysis, [55] highlight the need to: 1) use multiple news sources; 2) cover large geographical areas; 3) cover long time series; 4) enhance protest event analysis coding efficiency. The globality issues, i.e. the first three tasks, are successfully resolved by event databases created in the recent years, however, what still needs refinement is the coding procedure and quality. [20] also points out that there is a need of comprehensive data with good geotemporal coverage and coding runtime performance. Computational social scientists are constantly discussing the use of machine-coded event datasets, and come to a conclusion that the produced data is insufficient, often inaccurate, difficult to use, there are many improvements to be

done, such as ontology refinement and extension, named entity recognition enhancement, etc. [**?**].

1.1.1 Problem Definition

This thesis deals with the multilingual protest event data collection from news sources and annotation with event-specific features. No multilingual training corpora are available to train a feature classification model, and no outsourcing is currently possible. We understand our task as the creation of a prototype for protest event collection and information block parsing in a multilingual environment. Instead of translating the original articles into English, we apply multilingual event extraction methods. First, for a set of crawled and formatted headlines $H[h_1, h_2, ...h_n]$, we identify a subset $H_p[h_{p1}, h_{p2}, ...h_{pn}]$ related to protest events, and than, within H_p, we perform annotation of protest event type, location, reason, as well as duration, size, iteration, intensity and violence use, where mentioned. The output annotation of type *Protest_Event* thus includes slots for event ID, TYPE, LOCATION, REASON (POSITION and ISSUE), DURATION, SIZE, ITERATION, INTENSITY and VIOLENCE.

The obtained data can be used for protest activity monitoring and prediction. It can also contribute to the solution of resource credibility testing, which is the problem reported in most sociological papers related to protest activity studies.

By its nature the language of news reports provides the most important details of the message in the first lines to catch the attention of the reader. In case a new event is mentioned, the tabloids tend to provide a maximum sufficient description in the title, including event type, location, cause and other details. Our analysis is sentence-level and relies on the conceptual structure similarities in multilingual press headlines characteristic to these descriptions. In this work we consider several european Subject-Verb-Object languages: Bulgarian, French, Polish, Russian, Spanish and Swedish.

The hypothesis behind this work is that generic patterns based on the observed regularities in the conceptual structure can be effectively employed to extract slot-specific protest event data from multilingual press headlines with high precision and recall.

1.1.2 Objectives and Tasks

The objective of this thesis is to mitigate the problem of training data absence in multilingual protest event data collection and coding by creating the appropriate natural language processing tools. In order to build the system, the following tasks should be performed:

- Training data collection
  - Crawling
  - Filtering
- Concept hierarchy construction
- Study of the publicly available tools for the development and use of multilingual text processing components
  - Gate framework
  - PoS-taggers
    - Freeling (multilingual)
    - Treetagger (multilingual)
    - Mystem (Russian)
    - Stagger (Swedish)
- Multilingual corpus pipeline creation
  - Construction of generic JAPE pattern/rule pairs
  - Multilingual gazetteers building
  - Generic corpus pipeline building
- System output setting
- Performance of experiments on test sets

1.1.3 Terms

Since our work deals with the extraction of events, we should give an explanation of the event meaning adopted in the thesis. The multifaceted nature of the term across disciplines is shown, for instance, in [4][33]. In information extraction events are known to be more difficult to extract than entities and relations due to their representation complexity and to the unclearness of the analysis unit. There is no agreement about the exact representation of events across all languages and domains. An event is commonly perceived and defined as something that takes place and changes the state of affairs, therefore, it needs geographical and temporal attributes. Events are complex, because our perception "organizes" several or many happenings into a discrete event, as observed in [54]. A separate mention of a mass gathering or a speech for or against something does not necessarily denote a protest action. The eventhood of a sentence is often prompted by its informational structure, and the eventness of a word - by its context. In information

extraction, researchers widely use the representation of an event as a predication with arbitrary number of arguments and relationships. Event-related information is commonly represented as a report on who did what to whom, where and when, through what methods and why (ACE: `http://www.itl.nist.gov/iad/mig/tests/ace/`), with a varying number of components depending on the system task. The report is further stored as a database record (a scenario template with a pre-defined number of slots). According to ACE terminology, event trigger is the word that determines the event occurrence; argument is an entity mention, a value or a temporal expression that constitute event attributes and event mention is an extent of text with the distinguished trigger, entity mentions and other argument types [17][38]. In the present work we adopt this event definition.

Protest event is understood almost in the same way as it is in DoCA (`http://web.stanford.edu/group/collectiveaction/cgi-bin/drupal/`), a protest event database manually coded on the basis of the New York Times numbers issued during 35 years (1960-1995), as pointed out in [20]: protest events are (i) collective acts, (ii) public actions, (iii) protest actions, and (iv) events that make a specific claim about the need to change the state of affairs.

Generic informational pattern. In our vision, sentence event patterns are two-level. The lower level describes the syntactic chunk structure of the informational slots, and the upper level the informational structure of the sentence. In the context of protest events, informational patterns describe the allocation of the Event Type (Trigger), Reason, Location, Actor and other text blocks in a news headline. The patterns are considered generic, because they can be applied to a corpus containing texts in Spanish, French, Swedish, Bulgarian, Russian and Polish.

The rest of the definitions follow in alphabetical order:

Gazetteer is a single-column table that contains terms invoked by a text processing module. Usually, terms are separated with "$\backslash n$". Some software-specific gazetteer types support regular expression syntax.

Ontology is a formal description of a domain represented as a set of interconnected nodes, where the nodes are classes and subclasses of a concept hierarchy and the relations are object and data properties.

Pipeline is sequence of analyzers applied to unstructured data. Each analyzer produces a set of features, which can further be used for data classification or processing by the next analysis layers.

Semantic frame is a representation of a situation (e.g., voting, protesting) with its central element (predicate) and the participants (named entities, such as person names, organization names, location mentions, objects, and other conceptual roles) involved.

Tabloids are compact size newspapers that tend to emphasize sensational stories and cover a wide spectrum of political topics.

Template is the structure of event report stored in the database that commonly includes the slots for event type, location, date, main participants, victims, and other.

Treebank is a large corpus of text documents labeled with syntactic and semantic tags for further training of statistical models.

1.1.4 Road Map

The rest of the thesis is organized as follows.

- Chapter 2 gives an overview of up-to-date sociological and multilingual event extraction studies. Section 2.1 provides a general description of event extraction techniques. In Section 2.2, we outline the scope of modern sociological studies and define the place of automatic protest event data collection in sociological studies that use Internet as the main information source (Internet Sociology). Section 2.3 describes the state of the art in multilingual event extraction: systems and approaches they use focusing on the pattern-rule pairs construction.
- Chapter 3 explains in detail the order and parameters of system modules: web crawler, duplicate filters, Part-of-Speech taggers and GATE PRs.
- Chapter 4 gives details on the process of manual selection of event features that will be annotated together with the main event description (type, reason, location) and will support decisions on event relevance. In Section 3.1, the protest event concept hierarchy constructed on the basis of news headlines is presented. Its classes and subclasses are the necessary features that can be used by social scientists to automatically create a "map" for each of the protests events, thus structuring the data. In Section 3.2, we discuss the natural language representation of some of the event features (information slots) and present the corresponding JAPE grammars and gazetteers.
- In Chapter 5, the system is evaluated as follows. Firstly, we count the number of true positives, headlines crawled correctly for each of the languages. Secondly, we evaluate the obtained event descriptions in terms of Precision and Recall.

- In Chapter 6, we discuss the results, present some conclusions to the accomplished work and outline the plans for future studies and experiments.

Chapter 2

Overview of the Background

2.1 General Approaches to Event Extraction

An overview of general event extraction approaches is given in [24]. It addresses monolingual event extraction approaches implemented in the period from 1992 to 2010: knowledge-driven, data-driven and hybrid. The approach that relies entirely on linguistic pre-processing components (tokenization, sentence splitting, part of speech tagging, syntactic parsing, etc.), hand-crafted extraction patterns and knowledge bases (Wikipedia[1], WordNet[2], DBpedia[3], GeoNames[4], BabelNet[5], etc.) is called "knowledge-driven". It yields accurate results and requires no training data, but its disadvantage is in the need of a large effort of experts and human annotators, as well as time resources. Within this approach, lexico-semantic patterns map relationships from a knowledge base by detecting concepts on the basis of linguistic pre-processing and ontology-based dictionaries. Lexico-syntactic patterns operate on a less abstract syntactic level, the lexical level is represented by gazetteers that are not connected to a unifying knowledge base, which makes it difficult to handle paraphrases and other language phenomena. The pattern/rule pairs in the event extraction systems are commonly represented by a set of cascaded grammars (finite-state transducers), where each grammar layer builds on annotation offsets and features generated by previous layers.

The "data-driven" approach is commonly free of prior knowledge and uses unsupervised machine learning (e.g., clustering, expectation maximization, method of moments, principal component analysis, self-organizing maps, etc.), as well as other quantitative

[1] `http://wikipedia.org`
[2] `http://wordnet.princeton.edu`
[3] `http://dbpedia.org`
[4] `http://geonames.org`
[5] `http://babelnet.org`

techniques from probability modeling, information theory, linear algebra, etc. to make inferences from unlabeled data. Within this approach, the text document representation is based on simple word frequencies, TF-IDF (the frequency of a term in a given document multiplied by the inverse frequency of a term in a given collection of documents), n-gramms, etc.. Multilingual monitoring systems tend not to use unsupervised methods alone, because of a high level of inaccuracies and low interpretability of results.

The third class encompasses hybrid approaches that considerably proliferated in the recent years. They benefit from hand-crafted patterns and gazetteers, knowledge repositories, and quantitative techniques. Here, machine learning is used for pattern learning (weakly or semi-supervised approach) or the sequence classification of the event template slots on the basis of a labeled corpus or treebank, if available (supervised approach).

In [24], the approaches are evaluated with respect to the amount of training data, expert knowledge and expertise required, as well as the interpretability of results. The authors conclude that the hybrid approach is a compromise solution for building an event extraction system, however, the combination of techniques from different fields requires the corresponding level of expertise.

2.2 Protest Event Extraction and Coding

For decades sociologists from the institutions, such as Berkman Center for Internet and Society, University of Illinois at Urbana-Champaign Cline Center for Democracy and others have been studying regularities in contentious collective action and accumulating statistics for protest prediction and the analysis of its origins, dynamics and aftermath from protest event data (single events, small event sets and, since recently, big data). The advent of Internet and social media brought researchers to use the world wide "memory" to study the influence of social media messaging on the real-life protest actions, mutual influence between traditional mass media, often controlled by governmental authorities, and social media discussions, protesters' benefits of social media use, etc., as well as to improve the time-honoured manual political event data collection. News reports keep being the most used information source for protest event analysis. With the exponential growth of the Internet and the availability of digital news, cross-national studies get advanced by the second generation of PEA researchers in mid 1990s, as observed in [55]. The third generation concentrates on the news bias problem and pioneers in the use of electronic approaches: half-automated collection of political event data based on adapted versions of KEDS[6], and keywords use in order to speed up the search of relevant texts

[6] `http://eventdata.psu.edu`

in digital archives. The fourth generation looks into the activities within a single event, as well as the protest claim itself, and considers news bias less important.

Empirical studies of separate events often do well without computational methods: sociologists look into a protest in the thick of it by questioning the participants, reading and interacting in their online discussions, etc.. However, even for a single event much time is needed to complete the picture on its origins, consequences, actors involved, and parallels to other similar events. Retrieving and counting these features for lots of events across languages is too time-consuming, therefore, the development of the appropriate text analysis tools is the solution.

Researchers list the following advantages of automatic event data coding: ability to process large datasets in a short span of time, geotemporal coverage, replicability, modifiable dictionaries, no need to pay expensive human coders, objective view of the data. As drawbacks they mention the inability of the systems to parse complex sentence structures, metaphore, idioms, i.e. figurative language, and timedependent text [50].

Its main advantage is the public availability of texts for long time periods, however, there is a well-known drawback: bias. Today, events coverage differs from country to country and depends on state and newspaper politics: press may focus on local events, not to report on some specific country's events, or report on the same event differently than other newspapers. According to [20], news bias effects can be mitigated by comparing data from multiple multilingual information sources.

Kalev Leetaru, a co-creator of GDELT, observes in [16] that in the course of events collection for GDELT 70% of events reported in Portuguese, do not appear in English at all, thus highlighting the importance of covering local news in local languages. In fact, before the Google Translate scholarship, GDELT failed to detect the early mentions of Ebola in Guinea's news, because it was reported in French.

2.2.1 Manually and Automatically Coded Datasets

Most projects involving large-scale event data collection for sociological and other purposes do not focus particularly on the protest event, they consider it as one of the event types. In this Section, we examine several projects, and, as automatically coded datasets have similar workflows, we put focus on the comparative insights.

2.2.1.1 Manually Coded Datasets

Hand coding allows to collect complex qualitative and quantitative variables, and thus build a fine-grained description of an event, however, it is prone to subjective interpretation and costly in terms of time and expert resources.

The earlier mentioned Stanford University project DoCA (ongoing) manually gathered data on collective protest in United States on the basis of a single newspaper (The "New York Times"). Its codebook contains 38 positions, where a detailed description of each variable is given. For instance, there are 3 codes for the "Target" variable that constitute the answers to the following questions: 18a: "Was there a crearly definable target?", 18b: "If so, how many targets?", 18c: "If yes, what was the main target?". The use of such nuances may result quite complicated and even unnecesary. The main advantage of the dataset is that it provides quality data for a complete set of necessary protest event variables: event type, form, date, location, size, source actor, claim, target actor, initiating group, violence involvement, victims, which is already used to train and test the performance of the automatic English-language coders [20].

ACLED, the Armed Conflict Location and Event Dataset [41], is currently functioning as a supervised near-real-time processing system. It has a broader range of event types that are related to political violence. The term political violence denotes politically motivated violent attacks by people groups. The main procedure of event collection and coding is performed by experienced individual coders, and the results are consecutively reviewed by 3 experts. The algorithm captures event type, date, location, context, different types of actors, victims variables.

2.2.1.2 Machine-Coded Datasets

An automatically coded system aims at providing an unsupervised dynamically growing database of quantitative and qualitative features of different event types in a given domain. In the field of social protest events collection, all of the systems have a similar workflow. Events are detected on the basis of manually collected dictionaries, connected to domain ontologies of actors and events, thus, the quality of the protest event features depends on the underlying ontology. An EOI is a dyadic system, a relation between two entities, the Source Actor and the Target Actor. Each sentence of an article is considered as one or several events, where verb groups denote actions that trigger events, and the left and right noun phrases are the entities (actors). The main techniques used as the core algorithms are from Natural Language Processing and Machine Learning. Natural Language Processing techniques are quite superficial and include simple pre-processing

and shallow parsing. The granularity of the same features, such as the geotemporal unit, may differ. In case the system deals with non-English texts, they are translated into English using machine translation. In this and the following Section, we deal with the following projects that use computational tools: KEDS [43], El:DIABLO[7], W-ICEWS[8], SPEED [32], and GDELT [45]. The KEDS and El:DIABLO, the latter patronized by the Open Event Data Alliance, are ready-made and adjustable user-oriented systems for event data collection and coding. The SSP created within the framework of the SPEED Project of the Cline Institute for Democracy of Illinois is a queryable database covering specifically civil unrest-related events. W-ICEWS was developed for the needs of the US government by Lockheed Martin[9] and comprises a set of powerful computational tools for crisis data collection (iDATA), monitoring (iTRACE), forecasting (iCAST), and sentiment analysis from open-source social media (iSENT). It grew out of an earlier manually coded dataset. The access to SSP and iDATA is limited. GDELT was released in 2013 and constitutes the biggest open-source project in event data collection.

The general workflow of the systems based on machine coders is presented in Fig. 2.1. Some of the key features of the mentioned systems are shown in Tab. 2.1. The presented systems are modular and their performance depends highly on the state-of-the-art tools they incorporate.

The considered machine coding-based systems have a similar general pipeline. Feed aggregators (Fig. 2.1: Aggregator) collect news media articles from a whitelist of RSS (Fig. 2.1: Tabloids) related to a specific domain for a given time period. The resulting dataset undergoes the first-stage pre-processing, which includes total duplicates removal, formatting, and translation, in case the articles have not been pretranslated by news providers (Fig. 2.1: Filtering, Formatting, Translation).

The SPEED project uses historical news archives of the New York Times, the CIA Foreign Broadcast Information Service and BBC Summary of World Broadcasts that provide local press articles pre-translated into English. SPEED system includes a text categorization stage, where the articles related to politically motivated attacks, political expressions or state attacks are sorted out of the topically diverse dataset using BIN, a Naïve Bayes-based classifier.

At the second stage (Fig. 2.1: NLP pre-processors), the dataset is pre-processed using general-purpose NLP pipelines, such as Apache open NLP (for *tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and*

[7] `http://openeventdata.github.io/eldiablo/`
[8] `http://www.lockheedmartin.com/us/products/W-ICEWS/iData.html`
[9] `http://www.lockheedmartin.com/us/products/W-ICEWS.html`

coreference resolution) or the Stanford University CoreNLP (for *tokenization, lemmatization, sentence splitting, part-of-speech tagging, NER, dependency parsing, coreference resolution, sentiment detection.*

TABLE 2.1: The main characteristics of machine coding-based systems

Project	**KEDS**	**EL:DIABLO**	**W-ICEWS (2001-present)**	**SPEED (1945-2005)**	**GDELT 2.0 (1979-present)**
Supervision	Unsupervised/ Supervised	Unsupervised	Unsupervised	Supervised	Unsupervised
Customizable	+	+	-	-	-
Language Coverage	English	English	English, Spanish, Portuguese	English	Google Translate
Focus	Political interaction	Interstate conflict mediation	Interstate conflict mediation	Civil unrest	Interstate conflict mediation
Sources	Reuters, Agence France Presse	around 160 websites	over 6000 news feeds	over 800 news feeds	thousands of news feeds
Ontology	WEIS[10], CAMEO[11]	CAMEO	CAMEO	SSP	CAMEO
Dictionaries	KEDS[12] or other	customizable	own, entities and generic agents	-	WordNet[13] & NER-enhanced
Data acquisition	Nexis[14]	own web scraper	Factiva[15], OSC[16]	Heritrix[17], BIN[18]	Google Translate
Geocoder	CountryInfo.txt[19]	Penn State GeoVista project coder[20], UT/Dallas coder[21]	CountryInfo.txt	GeoNames[22]	CountryInfo.txt
NE &Event Coding	own sparse parser	CoreNLP[23], PETRARCH[24]	BBN Serif[25], JabariNLP[26]	Apache open NLP[27], EAT[28]	CoreNLP, PETRARCH

[10] See [31]
[11] See [44]
[12] KEDS (WEIS coding for Middle East), Pevenhouse (Behavioral Correlates of War, coding system for the Middle East), PANDA Project dictionaries[44]
[13] WordNet thesaurus: http://wordnet.princeton.edu
[14] Public Nexis Search Engine: *http://lexisnexis.com*
[15] Commercial Factiva Search Engine: *http://new.dowjones.com/products/factiva/*
[16] *http://en.wikipedia.org/wiki/Open_Source_Center*
[17] The Internet Archive's open-source crawling project: *https://webarchive.jira.com/wiki/display/Heritrix/Heritrix*
[18] Automatic Document Categorization for Highly Nuanced Topics in Massive-Scale Document Collections [28]
[19] CountryInfo.txt includes about 32 000 entries on 240 countries and administrative units: country names, synonyms, major city and region names, national leaders:*https://github.com/openeventdata/CountryInfo*
[20] Geovisualization and Spatial Analysis of Cancer Data:*http://www.geovista.psu.edu/grants/nci-esda/software.html*
[21] Geospatial Information Sciences (GIS) project by the University of Texas at Dallas: *https://github.com/mriiiron/utd-geocoding-locator*
[22] Free Geographical Database:*http://www.geonames.org/*
[23] Java-based NLP tools including conditional random fields-based NER: *https://github.com/stanfordnlp/CoreNLP*
[24] Open Event Data Alliance Software: *https://openeventdata.github.io/*
[25] See [6]
[26] See footnote 25
[27] A maximum entropy and perceptron-based machine learning NLP toolkit: *http://opennlp.apache.org/*
[28] See [22]

The next stage (Fig. 2.1: Machine Coding) is the proper event coding with PETRARCH, JABARI-NLP or TABARI. Currently, both TABARI coding engine [47] and its NLP-enhanced Java version JABARI-NLP [49] tend to be replaced by the state-of-the-art PETRARCH. The coding engine uses as input a collection of texts analyzed with Penn Treebank[29] or the previosly mentioned NLP toolkits. Main event slots are annotated using syntactic patterns, unless labeled on the previous stage by NLP pre-processors, and the resulting event types are matched against an event-actor ontology codebook (CAMEO, SSP or other) (Fig. 2.1: Event-Actor Ontology) on the basis of event-actor dictionaries (Fig. 2.1: Event-Actor Dictionaries).

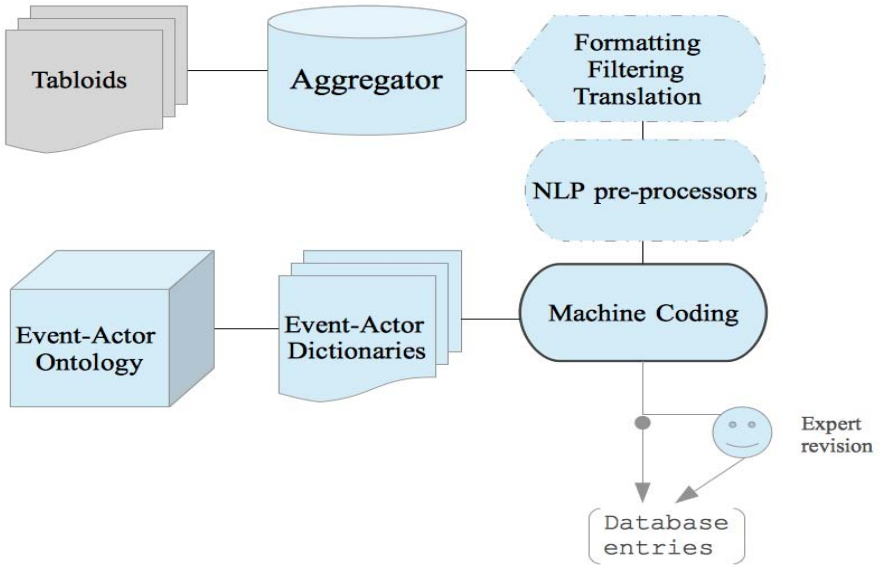

FIGURE 2.1: A sample pipeline of machine coding-based projects

W-ICEWS dictionaries contain data on actors (proper names) with alternate spellings and agents (unnamed entities like "insurgents", "students") with synonyms and their time-dependent affiliations[30]. As for the coding, W-ICEWS project uses a mixed approach: JABARI-NLP for shallow parsing and BBN's Serif to extract Entity-Relation-Entity triples as the coding basis. BBN Serif (proprietary) tool employs a hybrid of machine learning and pattern-based approaches for evidence extraction.

After coding and prior to database storage, event reports undergo final deduplication in some pipelines, e.g. in Phoenix (Open Event Data Alliance) used in El:DIABLO. GDELT is reported not to remove duplicate articles. Consequently, it measures reporting coverage rather than events, which is discussed on the leading social science pages, such as Bad Hessian blog[31], Jasper Ginn and Jay Ulfelder's blogs, in [20], etc.. In SPEED, all

[29] https://www.cis.upenn.edu/ treebank/
[30] http://thedata.harvard.edu/dvn/dv/icews
[31] http://badhessian.org/

of the resulting entries are manually corrected and filtered by the experts before storing them into the database (Fig. 2.1: Expert revision).

2.2.2 The Core Event Coding Algorithms

Event selection and event coding are two main subtasks within the computational approach to protest database population, where, giving a rough definition, *event selection* is the automatic collection of relevant articles describing events in a given domain or domains, and event coding is the procedure of automatic code assignation to certain pieces of information: for instance, a code denoting a governmental actor can be similar to **[GOV]**. Event selection algorithms for KEDS, El:DIABLO, W-ICEWS, SPEED, and GDELT are briefly described in the previous section. They rely on keyword-based queries and standard topic categorization approaches. This section focuses on the process of event coding. Here, we consider the CAMEO-based coding with PETRARCH and BBN's Serif, and the SSP-based coding with EAT.

2.2.2.1 Ontologies

CAMEO

CAMEO ontology is being developed since 2000. It focuses on interstate conflict mediation and originates from Cold War event ontologies (WEIS and others). Therefore, it does not capture some politically relevant events, such as crisis events, criminal and financial activity, migration, refugees, human rights violation, electoral and parliamentary activity [46]. It includes 20 Tier-I action verb (event triggering) categories: 01: Make Public Statement, 02: Appeal, 03: Express Intent to Cooperate, 04: Consult, 05: Engage in Diplomatic Cooperation, 06: Engage in Material Cooperation, 07: Provide Aid, 08: Yield, 09: Investigate, 10: Demand, 11: Disapprove, 12: Reject, 13: Threaten, 14: Protest, 15: Exhibit Force Posture, 16: Reduce Relations, 17: Coerce, 18: Assault, 19: Fight, 20: Engage in Unconventional Mass Violence [44]. [48] point out that this ontology is not aimed at coding contentious collective action. Under "14: Protest" position it lists only 4 types of protests events, detailed with some of the main protest targets and forms in the CAMEO manual [44]: 140. Engage in political dissent, 141. Demonstrate or rally (with the Tier-II categories: demonstrate or rally for leadership change; for policy change; for rights; for change in institutions, regime), 142. Conduct hunger strike (same Tier-II as in 141), 143. Conduct strike or boycott (same Tier-II as in 141), 144. Obstruct passage, block (same claims as in 141), 145. Protest violently, riot (same Tier-II as in 141). 140 is a general subcategory that does not specify any protest form: all demonstrations against a target actor. Support demonstrations will be

coded separately: possibly, within the 05: Engage in Diplomatic Cooperation category. Verbal protest expression belongs to a different category, because it is considered less forceful than mass gatherings. Military protests enter the 19. Fight category. Other events or attributes that are often related to protests, such as support on behalf of an entity, threat, and others, are distributed throughout the ontology. No temporal or other relations between events are specified, therefore, subevents will not be associated with the main protest event they are attributed to in the natural text.

The actor ontology lists state, as well substate agents that include police, military forces, social movement organizations, unaffiliated protest groups, etc.. As observed in [20], Tier-II categories of the 14: Protest category are rarely used in practice. Also, CAMEO actor specification is relevant for interstate relations and much less for the interactions between state and society or social movements.

SSP

SSP created within the framework of the SPEED Project of the Cline Institute for Democracy of Illinois is a direct entry database that consists of six main sections (who, what, how, where, when and why), each of which is supported by a number of questions that allow users retrieving necessary data on an EOI. The protocol contains a detailed ontology of destabilizing events: 65 Tier-I and Tier-II categories and over 100 categories in total. A destabilizing event is defined as a happening that unsettles the routines and expectations of citizens, causes them to be fearful, and raises societal anxiety about the future [32]. The database encompasses texts collected from the newspapers issued in 165 countries in the Post WWII era. All the articles are pre-translated into English. The final protocol design iteration includes eleven sections responsible for data processing. The sixth section deals with the domain ontology of event types consisting of three main Tier 1 categories (political expression events, politically motivated attacks, disruptive state acts). Political expression involves an obligatory presence of such parameters as public articulation, non-governmental actor, threatening or unwelcome political message. The main expression modes are a) verbal or written message, b) symbolic act, c) forming an association and d) mass demonstration or strike with the subsequent subcategories. Politically motivated attacks are violent actions or attempts by non-governmental initiators. Political motives in this case are defined as hatred toward socio-cultural groups or revenge for their prior actions, desire to change or control the government, follow or oppose a political ideology, advance a social cause etc.. Disruptive state acts include extraordinary or repressive acts by governmental initiators.

SPEED personnel constructed the ontology by exploring the literature on political violence, terrorism, political instability, and social movements in search of event

categories. Secondly, real data was analyzed and the respective classification was refined. Event-specific information that is relevant to the study of the origins and development of civil unrest was defined. Event attributes, such as geospatial and temporal information (event coverage, latitude/longitude, precise/estimated time and date), participants (governmental/non-governmental initiators and their traits (number, weapons used), messengers, rioters, reactors), consequences (negative/positive to initiators and actual participants), targets and effects (what happened to whom: damage, injuries etc.), origins (why it happened) and event linking, are distributed between the other sections [11] [22] [32].

2.2.2.2 Event Annotation and Coding Software

PETRARCH

PETRARCH is a successor of the TABARI coder. TABARI performs Entity-Relation-Entity triples annotation over shallow syntactic parsing. Its pattern-based approach is quite robust, however, it produces many false positives due to ambiguity issues. TABARI is a successor of KEDS, developed during 1990s. Currently, only PETRARCH and its dictionaries of actors, agents and verbs[32] are being extended and improved. The distinction of PETRARCH consists in the use of a full parse input from Penn TreeBank instead of false positive-prone, but robust, pattern-based shallow parsing [47] and the recent CAMEO.verbpatterns.140609.txt dictionary that includes both parser-based matching and extensive synonym sets. However, the runtime performance still suffers from the innovation. The previous system was able to code from 1000 to 2000 sentences per second depending on the dictionary, and the current one - around 150 sentences per second. CoreNLP parses around 2-5 sentences per second. The main problems that still arise after the modernization are as follows: 1) word sense disambiguation errors; 2) syntactic structure compression, which is encountered each time more often in the written language of news reports yields parser errors; 3) inability to handle sentences with complex syntactic structure. The authors see the following advantages of the modernization: 1) potential ability to handle multilingual news; 2) dictionary-independent identification of actors through noun phrases; 3) an improvement of source actor identification due to the correct classification of direct objects; 4) noun/verb/adjective disambiguation; 5) PETRARCH is written in Python, which is more young and flexible than TABARI's C/C++. A sample coding of a sentence relevant both for TABARI and PETRARCH (the image is taken from [46]) is shown in the Fig. 2.2.

[32] https://github.com/openeventdata/Dictionaries

BAGHDAD. Iraqi leaders criticized Turkey on Monday for bombing Kurdish militants in northern Iraq with airstrikes that they said had left at least one woman dead.

Event Code: 111
Source: IRQ GOV
Target: TUR

Event Code: 223
Source: TUR
Target: IRQKRD REB

FIGURE 2.2: A sample coded sentence (TABARI & PETRARCH)

As it can be seen, the coder provides data on the source and target actors' locations and roles, as well as the CAMEO code of the event. Here, **REB** - rebel, primary role code, and **GOV** - government, primary role code. Primary role codes are generic role codes. Primary roles are used for domestic actors. **REB** stands for armed and violent opposition groups with unclear aims. **TUR**, **IRQ** and **IRQKRD** (the actor domestic country and region specification) are entities' geolocations, although in our 2012-year version of the CAMEO manual we find only the name for the ethnic group coded as **KUR**. The event 111 (Criticize or denounce) belongs to the category 11 (Disapprove). As for the "bombing" event, our manual does not address any event 223 in the category 20 (Engage in Unconventional Mass Violence), rather because the 2012 manual version does not contain the recent improvements (2012-2015).

Essentially, the coder checks all the verb phrases in each clause of the sentence and in case no actors are designated after dictionary lookup, source and target are identified on the basis of the parsed input. The source actor is the first actor (noun phrase) in the sentence. The target is the first actor (noun phrase) on the right side of the verb that has a different code. In case, there is no actor on the right, the left one is considered the Target.

EAT

EAT developed by Quang Xuan Do, a member of the Cognitive Computation Group at the University of Illinois (past), was introduced into the SPEED pipeline in 2014. It employs a machine learning approach based on tens of thousands of manual event annotations (event, date, location, actor triggers) by trained human coders to build cognitive NLP-based models for automatic event data annotation within raw news text.

They observe that this tool is geared towards the mitigation of problems with event selection and duplicates removal [9] [22]. The SPEED project papers do not specify the machine learning techniques used.

Serif

BBN's (Bolt Beranek and Newman technologies) Serif is the state-of-the-art language-independent system for Entity-Relation (Predicate)-Entity triples (including ontology-based) extraction. The system uses an experimental combination of statistical algorithms that learn event models from manually annotated datasets in potentially any language. Obviously, it requires a large amount of expert annotations.

The distinctive feature of the approach is the intermediate representation of text in the form of the underlying propositions as an alternative to effort-intensive and error-prone parse trees. The propositions are formally described as: predicate $(role_1 : arg_1, role_2 : arg_2 ... role_n : arg_n)$, where predicate is a verb or noun, which differs from the standard event representation predicate $(arg_1, arg_2 ... arg_n)$ by the presence of the assigned roles. Relations are collected from propositions that summarize the literal meaning of the text. From the phrase "government was attacked by the party" the model learns the logical subject ("party") and logical object ("government") of the proposition. Arguments can be represented by not only entities, but also other propositions. The most common roles are: logical subject, logical object, noun phrase premodifier, predicate modifier (object of a prepositional phrase).

Another distinctive feature is the mixed use of learning models to improve the accuracy of the triple extraction. The result achieved by the best-performing model goes to the output. The first combined model that contributes to the final classification includes a generative model and a features-based model. The generative model calculates the probability of each propositional structure to represent a specific relationship, which is the joint probability of the relation and the structure. The respective probabilities are calculated as a smoothed mixture of maximum likelihood estimates. The feature vector model (similar to the pattern-based approach) represents the triple using 5 features: the predicate, and the entity types with their syntactic roles. The best fitting relation type is determined using a maximum likelihood metric. The third approach builds on discriminative models (voted perceptron training algorithm) and takes the inter-dependent features into account. There are 11 templates composed from atomic features: 1) types of entities or their mentions, 2) aspect of the predicate-argument structure, 3) predicate stem, 4) string value between two mentions, 5) WordNet synset of the predicate, 6) word cluster of the predicate [6]. The features 4 and 5 allowed this algorithm to outperform

by 3.2% (the authors do not specify the evaluation metric) the combination of the previous ones in many cases. The authors do not address the relation types used for the experiments.

2.2.3 Recent Advances in Protest Event Selection and Coding

This Section focuses on the protest as EOI. Here, we explain two machine learning and NLP-based approaches to automatic protest event selection and coding implemented by sociologists in the recent years.

[55] propose a computational linguistics approach to deal with the problems of protest data postselection after simple keyword-based crawling with LexisNexis, and coding. LexisNexis query for protest event is presented in Fig. 2.3 (the truncation (!) is set to retrieve various word endings; I/1 means that the search terms are within 1 word from each other). The news come from "The Guardian" (2010 archive). As it can be seen, the query covers many types of contentious collective action. There are not only protests against some actor, but also actions that have other purposes, such as support, tribute to one's memory, etc.: *signature collection, petition submission, protest, demonstration, manifestation, march, parade, rally, picket, riot, human chain, affray, letter campaign, festival, ceremony, street theater, road show, student strike, vigil, boycott, hunger strike, blockade, sit-in, squatter, mutiny, assaults (colour, fire), attack, bombing, arson with synonyms, fire raising, sabotage*, and other. As it may be expected due to the terms ambiguity, this search gives lots of false positives (only 68 out of 727 crawled articles address protest events).

Each text undergoes a complete linguistic preprocessing: tokenization, stopwords removal, lemmatization, PoS tagging[33], sentence splitting, and dependency parsing[34]. Parsing proved to make no effect on the results of the classifier, however, authors consider its application useful for the event coding. For instance, dependency triple "protest + against + cuts" extracted from the sentence "Workers protest against the Government's proposed public sector pensions cuts" indicates the claim of protesting groups.

For the postselection, an active learning binary classifier is built, which uses the output of the java-based UIMA framework[35] and hidden topic models[36] as features. Active learning implies an active involvement of the expert/annotator in the learning process. At the first stage, the classifier is learned from a small collection of labeled texts.

[33] TreeTagger (http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/) and Penn Treebank tagset (http://www.cis.upenn.edu/ ?treebank/)
[34] Mate tools: http://code.google.com/p/mate-tools/
[35] https://uima.apache.org/
[36] Mallet toolkit: http://mallet.cs.umass.edu/

FIGURE 2.3: A LexisNexis query for protest events retrieval

```
((submission OR submit! AND initiative OR referendum) OR petition! OR
    (collect! AND signature! AND campaign!) OR protest! OR demonstrat! OR
    manifest! OR marche! OR marchi! OR parade OR rall! OR picket! OR
    (human chain) OR riot! OR affray OR (letter! I/1 campaign!) OR parade
    OR festival OR ceremony OR (street theatre) OR (road show) OR vigil OR
    (consumer OR lecture OR university OR campus OR college OR school OR
    pupil! OR student! AND strike!) OR boycott! OR (hunger strike!) OR
    blockade OR (block! AND street OR traffic OR area OR site) OR sit-in
    OR (sit! AND strike!) OR squatter! OR (squat! AND house OR building OR
    area OR property) OR mutin! OR bomb! OR firebomb! OR molotov OR
    graffiti OR (paint! OR colour OR fire AND assault) OR attack OR arson
    OR incendiar! OR (fire I/1 raising) OR (set AND ablaze) OR landmine OR
    sabot! OR hostage! OR assassinat! OR shot OR murdered OR killed
```

Secondly, a large unlabeled corpus is processed, on the basis of which the algorithm decides, what units should be labeled next by the expert/annotator to improve the classifier performance. Basically, the algorithm chooses the most "useful" samples that are unknown to it to reduce learning time expenses and increase efficiency. In practice it means that for the chosen samples the conditional distribution of the target class has the highest entropy, given certain feature values [38]. The evaluation of the classification algorithm has been carried out on the training and development sets with the reported performance of 88.6% (Recall) and 70.1% (Precision).

As for the coding procedure, the system is designed to extract 6 main event features: **protest issue** (economic, cultural, security/peace, institutional/campaign), **protest forms** corresponding to each of the issues (petitions, letter-writing campaigns, and others (economic issue), public demonstrations (cultural issue), strikes/boycotts/occupations (security/peace issue), attacks/blockades (institutional/campaign issue)), **number of participants, location, date**, and **participating organizations**. The identification of all of them but issue and form is covered by the Stanford NER tool. Only the number of participants feature required the construction of a separate gazetteer. Protest issue is supposed to be identified by measuring the distance between a given text and the corresponding hidden topic. Protest forms are supposed to be identified using heuristic rules on the word space models output (dependency triples). No evaluation of this approach is presented.

In [20], preliminary experiments are conducted to introduce a supervised learning-based system for protest event data collection and coding. Two SVMs (Support Vector Machine classifiers) are trained on the "The New York Times" articles (DoCA dataset) using Python scikit-learn in order to (1) perform event selection, or distinguish between protest and non-protest articles (binary classification), and (2) perform partial event coding (first coding stage), or classify claims, targets, dominant protest forms and initiating groups for each of the protests (multiclass classification). The third task (second coding

Table 2.2: Protest event selection & coding systems overview

System	**Wueest et al., 2013**	**Hanna, 2014**
Language Coverage	English	English
Training Set	"The Guardian"	"The New York Times"
Pre-processing Toolkit	UIMA, HTM	-
Concept Hierarchy	-	DoCA
Postselection Algorithm	Active Learning	SVM (binary mode)
Coding Algorithm	NER & heuristics	SVM (multiclass mode)

stage) and its solutions are traced, however, it is reported to be a work in progress, and no evaluation results are presented. It consists in the classification of location, event size and organizations involved by training a linear-chain conditional random fields-based named entity recognizer of the Stanford NLP group.

Pre-processing includes XML formatting, lowercasing, punctuation and stopwords filtering, TF-IDF text representation on the basis of the original word forms (no stemming). Classification options are set to linear kernel and soft margins to better handle imbalanced classes. Documents are represented using TF-IDF counts of unigrams.

The binary classification demonstrates a very low precision value (0.06) for the protest class, and, having analyzed the results, the author proposes roughly to use a margin value (1) to distinguish between protests (above 1) and non-protests (below 1). Multiclass SVM produces a balanced classification, where the Claim and Target classification Precision and Recall are around 70% (arguably, because of very clear natural language indicators of these concepts), the Initiating Group - 61% and 54% respectively. The protest Form classification lags behind with the values of Precision and Recall under 50%. The Target classes are generalized on the basis of the DoCA specifications as follows: Government/State, Private/Business, University/School, Medical Facility/Organization, Other, Ethnic/Racial Group.

The proposed approach is quite natural in the context of the overall integration of machine learning techniques into the social science, however, its use in this particular case is accompanied by a number of difficulties due to the training set incompleteness (monolingual, limited news sources, etc.). The author outlines the following refinements to be implemented: 1) extend the coverage of news sources and annotate the obtained corpora in order to feed the classifier (because of DoCA's blind spots, limited vocabulary, "The New York Times" bias); 2) ensure the extraction of multiple events (protests or other) per article as in PETRARCH.

The approaches of the overviewed systems for protest event selection and coding are summarized in Tab. 2.2.

2.3 Multilingual Event Extraction Approaches and Systems

The systems for multilingual news monitoring tend to use hybrid approaches to event extraction. They are highly modular and incorporate lots of language-dependent tools for pre-processing and pattern recognition. The present Section extends the overview of multilingual event extraction systems given in [11]. Multilingual processing has been recognized by information extraction researchers as key functionality for complementarity and less biased view of events and opinions [51]. Cross-lingual information fusion improves the quality of event extraction performance [37]. Event extraction systems currently extend their language coverage in order to monitor the emergence of specific events in target regions, where valuable data appears in local languages. One of the approaches used earlier in multilingual information extraction systems is MT-based document representation followed by the application of monolingual information extraction methods. For instance, MURASAKI system, developed by Systems Research and Applications Corporation in 1994 [3], uses an internal language-neutral text representation (interlingua). MIETTA (1998-2000) [8] for tourism and travel assistance (in five languages) also uses MT and English as an interlingua for multilingual thesaurus construction. Also, it applies scenario templates for IE and natural language generation. In this approach we already have all the data in an intermediate form and there is no need of the further information fusion. The main drawback of MT use is the quality of translation, e.g. of proper names, domain-specific and context-dependent terms and collocations. State-of-the-art approaches consider filtering and processing text in the original language [51] with the further event merging in a database as a more reliable strategy.

Multilingual event extraction systems mine different types of data sources, which may or may not contain metadata mark-up: media (news articles, blogs, twitter), reports (SIGINT, HUMINT, etc.), Wikipedia, personal information (social profiles, CVs, private documents), domain-specific web pages (e.g., tourism-related for recommender systems). The commercial system NetOwl[37] uses a blackbox technology to analyse cyber security-related events from incident reports and works with eight languages (English, Chinese, French, German, Korean, Persian, Russian, Spanish). Biographe project[38] extracts person-related facts in four languages (English, Danish, German and French) from social profiles, CVs etc.. MULTISENSOR (Mining and Understanding of multilinguaL contenT for Intelligent Sentiment Enriched coNtext and Social Oriented inteRpretation) is an upcoming EU-funded project being geared towards domain-independent business decisions support. Once complete, the system will be capable of analyzing diverse multilingual content, including TV, radio emissions and WWW in four languages (English,

[37] http://netowl.com
[38] http://biographe.org

Spanish, French, and German). MULTISENSOR multilingual text processing techniques are essentially similar to those considered in detail in what follows in this Section: a pipeline for NER, concept recognition based on multiple lexical resources (BabelNet[39], Unified Verb Index[40], Eurovoc[41], Reegle Glossary[42]), relation extraction pipeline (deep dependency parsing and semantic role labeling). NER is implemented in Java and includes several stages: text segmentation into sentences, tokenization and NER itself (local parser, text analyzer (coreference resolution) and output generator). Named entities are annotated with Brat[43], a web-based annotation tool [2].

The general pipeline of multilingual event extraction systems is shown in Fig. 2.4. It is similar to those of the previously considered systems for socio-political event extraction.

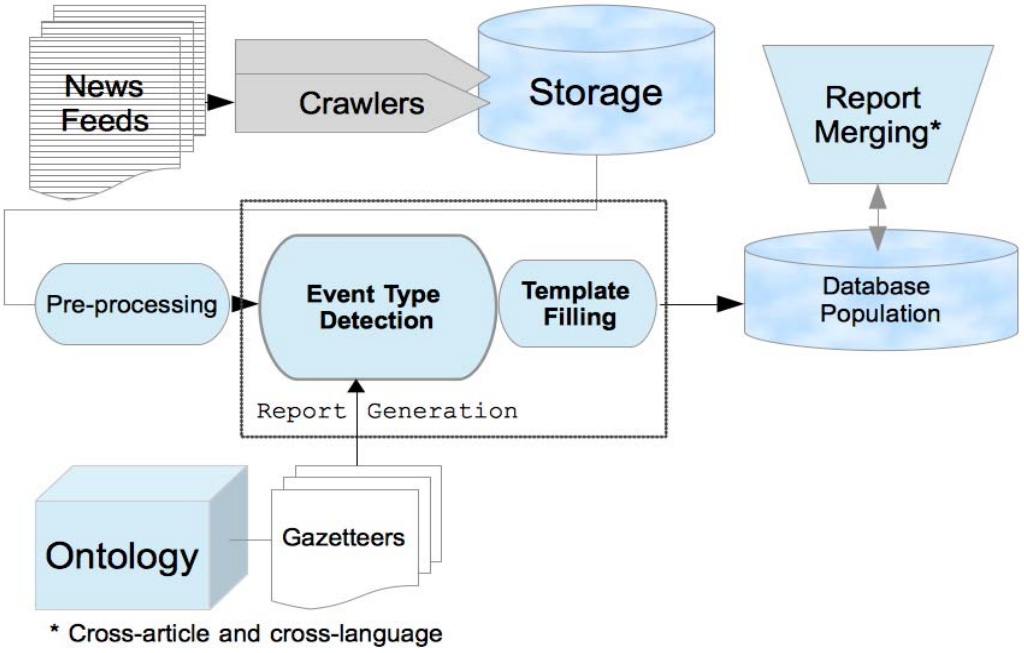

FIGURE 2.4: Multilingual event extraction system sample workflow

News feeds are first mined to extract the title, text body, publication date, author name, etc. of the target articles, and the obtained information is saved to a storage. Secondly, news articles undergo a primary pre-processing (from tokenization to basic NER). At the next stage, the event type is detected using ontology-based gazetteers and, finally, the rest of the template slots are filled, such as the type and number of participants, organizations involved, the type and number of victims, latitude and longitude of a given event location, time and date of a given event, weapons used, etc..

[39] http://babelnet.org/search.jsp
[40] http://verbs.colorado.edu/verb-index/search.php
[41] http://eurovoc.europa.eu/drupal/
[42] http://www.reegle.info/glossary
[43] http://brat.nlplab.org/index.html

The following multilingual event extraction systems are scrutinized in the present Section: NEXUS, NewsReader, PULS, Xlike, ZENON. The choice of these systems is due to the availability of open access details on their architecture and the fact that these have been some of the main projects in the multilingual event extraction domain. The general information on the systems is given in Tab. 2.3: language coverage, domain of focus, data sources and analysis unit.

TABLE 2.3: General description of multilingual event extraction systems

Project/Tool	**NEXUS**	**NewsReader**	**PULS**	**Xlike**	**ZENON**
Language Coverage	English, Italian, Spanish, French, Portuguese, Russian, Arabic	English, Dutch, Italian, Spanish	English, French, Russian	English, German, Spanish, Chinese, French, Italian, Portuguese, and Dutch	English, Dari, Tajik
Focus	European cross-border activity, natural disasters	Finance and economics	Epidemic threats, European cross-border activity	General	Interaction within military deployments
Sources	News media	News media	News media	News, blogs	KFOR corpus
Analysis Unit	Lead sentences of each article in a news cluster centroid	News article (full)	News article (full)	News cluster centroid	HUMINT report (full)

NEXUS [5][37], a project of the Joint Research Center of the European Commission (1999-present), is a tool developed for Frontex (the European Agency for the Management of Operational Cooperation at the External Borders of the Member States of the European Union) to support the surveillance of European borders and third countries, and intelligence-based risk analysis. Event extraction engine has been developed in collaboration with PULS being addressed later in this Section. The systems differ in the way they look at the text and in language coverage. They are complementary, because NEXUS performs cluster-centric shallow analysis of news head sentences, and PULS - an article per article deep linguistic parsing, which allows them to retrieve sets of event types that only partially overlap. NEXUS is focused on events, related to illegal migration, cross-border crime, and crisis situations at and beyond EU borders, and in third countries. Active language coverage includes English, Italian, Spanish, French, Portuguese, Russian, Arabic. According to the authors, the system adaptation to a new language takes around 3 months.

The input for the NEXUS engine, as well as for PULS, includes collections of news articles supplied by the EMM[44] news aggregator and clustered according to their content similarity, each of the clusters being geolocated and meta-data tagged. Cross-border security-related event clusters are sorted out using a keyword-based heuristics. More specifically, according to [5], the EMM functionality consists in the following:

- Scraper: checking for RSS updates;

[44] http://press.jrc.it

- Grabber: creating a new RSS feed for each website with the title, link, description, and text;
- Language detector: identification with automatically populated word frequency tables;
- NER: people and organization detection with an in-house multilingual automatically populated information base;
- Geocoder: location detection with GeoNames and an in-house gazetteer, disambiguated before and after clustering;
- Topic categorization: category assignment basing on multilingual regular expression patterns and boolean logic;
- Meta-data filtering: filtering with expressions that use boolean logic over meta tags;
- Clustering: average group linkage agglomerative clustering with the bag of words method (and some rules: ignore 100 top frequent and two-character words) used to create vector features.

The resulting clusters belonging to Frontex categories are subject to further processing by NEXUS and PULS. NEXUS is aimed at scanning only the lead sentences and title of each article in a cluster, and detecting one event per cluster. To this end, the system performs shallow parsing, simple cascade finite-state grammars application, followed by cross-article cluster-level report merging. Firstly, articles are linguistically pre-processed with an in-house toolbox, CORLEONE (Core Linguistic Entity Online Extraction), including tokenizer, sentence splitter, dictionary lookup (numbers, quantifiers, person titles), unnamed person groups extractor, and morphological analyzer. Secondly, a cascade of finite-state transducers encoded and compiled with another in-house tool, ExPRESS (Extraction Pattern Recognition Engine and Specification Suite), is applied. Patterns are represented by regular expressions over flat feature structures (non-recursive with string attributes). The target report contains the following slots: TYPE, SUBTYPE, DESCRIPTOR (free text), SNIPPET (triggering fragment), PUB_DATE (publication date), DATE (event date mention in the text body), LOCATION, CONFIDENCE (an automatically determined value of system's confidence in a selected event), RELEVANCE (an automatically determined relevance of an event to the end user), SEVERITY (event severity depending on the number of victims, injured, etc.), SOURCE, PERPETRATOR, VICTIM, ITEM (contraband goods), MEANS (transport means used to cross the border), NUM_AFFECTED (number of affected), NUM_INJURED, NUM_KILLED,

FIGURE 2.5: A sample ExPRESS rule

```
killing_event :> ((person_group & [NAME: #n1,
                                   AMOUNT: #a1,
                                   QUANTIFIER: #q1]
                  |person & [NAME: #n1,
                             AMOUNT: #a1]):victim
                 (dictionary & [TYPE: "death_trigger",
                                FORM: "passive"
                                METHOD: #m])
                 ((person_group & [NAME: #n2,
                                   AMOUNT: #a2,
                                   QUANTIFIER: #q2]
                  |person & [NAME: #n2,
                             AMOUNT: #a2]):killer

-> killer: actor & [NAME: #n2,
                    AMOUNT: #a2,
                    QUANTIFIER: #q2],
   victim: dead & [NAME: #n1,
                   AMOUNT: #a1,
                   QUANTIFIER: #q1]
                & IsNonZeroQuantifier(#q1),
   event: violent_event & [TYPE: "killing",
                           METHOD: #m,
                           ACTOR: #n2,
                           VICTIM: #n1].
```

NUM_ARRESTED, WOMEN/MINORS_INVOLVED (whether women or minors were involved in a cross-border activity).

ExPRESS is based on JAPE and borrows some of the features and syntax (e.g., availability of functional operators on the right-hand side of the rules) from XTDL, a SProUT platform formalism[45], which is similar in its fashion to JAPE as well. ExPRESS main components are a grammar parser and an interpreter of cascaded grammars.

A grammar rule consists of a left-hand side (LHS) that describes pattern constraints and a right-hand side (RHS) with annotation manipulation commands. Patterns include 1/2 slots (semantic roles) with string attributes and match simple syntactic constructions. In Fig. 2.5, a sample rule to detect violent events is presented (from [36]). The RHS (before the "->") matches the following sequence: "person/person group as victim (this component uses the annotation types "person" and "person_group" introduced in a previous phase), event trigger (this component uses triggering terms dictionary lookup) and person/person group as killer or actor (this component also uses "person" and "person_group" types introduced in a previous phase).

[45] http://sprout.dfki.de/

The LHS produces a template for the three main labels (event type together with victim and killer roles). The sentence "Most of the 230 Talibani were shot by the US troops" will be converted into the following scenario description:

```
dead & [NAME: "Talibani", AMOUNT: "230", QUANTIFIER: "Most of"]
actor & [NAME: "US troops"]
violent_event & [TYPE: "killing",
                 METHOD: "shooting",
                 ACTOR: "US troops",
                 VICTIM: "Talibani"].
```

NewsReader[46] (2013-2016) is an upcoming tool for multilingual event extraction, tracking, and analysis within the financial and economic domains. It parses four languages (English, Dutch, Italian and Spanish), using learned and ready-made preprocessing models and multilingual knowledge bases. The workflow includes some common steps, such as sentence segmentation, tokenization, part-of-speech tagging, lemmatization, NER, Syntactic Parsing and Coreference Resolution, performed by the Java-based IXA pipeline for Spanish and English[47], Alpino parser[48] for Dutch and corpus-trained modules for Italian. NewsReader developers also added such modules as Semantic Role Labelling (for English and Spanish), Named Entity linking to Wikipedia and DBpedia[49], TimeML annotation (based on James Pustejovsky research [40]), event classification, factuality, discourse and opinion mining to the basic functionality, which is relevant mostly for the English component, as it is the case in most multilingual information extraction systems. The other components have relatively limited functionality, due to the absence of knowledge resources and annotated corpora. Factuality is an indicator of whether an event actually took place, calculated using textual and structural indicators (modal verbs and other). Also, the authors added the indicators of Authority and Trust of an article measured using webpage metadata and stylistic properties of text.

The core strategy behind this event extraction engine consists in integrating the knowledge from WordNet and VerbNet (and similar predicate databases, such as FrameNet and PropBank)[50] to draw events out of predicates and their semantic role maps. However, the adaptation of predicate databases to new languages is very labour-intensive, time-consuming and requires much expert linguistic knowledge. In the absence of semantic role labeling functionality, Dutch and Italian modules make use of dependency parsers.

[46] http://www.newsreader-project.eu/
[47] https://github.com/ixa-ehu
[48] http://www.let.rug.nl/vannoord/alp/Alpino/
[49] http://dbpedia.org
[50] http://verbs.colorado.edu/semlink/

The event annotation engine used, NLP Annotation Framework (NAF)[51] is an offspring of KAF, KYOTO/Knowledge Annotation Framework[52], a multilingual semantic annotation tool. These tools are geared precisely towards multilingual semantic annotation, and already integrate WordNet, ontology (KYOTO-DOLCE), and fact generation support. Event retrieval is performed on the sentence level on the basis of 3 predicate classes (Communication, Cognition and Other) with their roles. The "Other" class denotes domain-specific actions. The corresponding predicate maps are taken from the predicate matrix, a special resource that integrates predicate data from FrameNet, VerbNet, PropBank and WordNet [1].

PULS[53] has been developed in the University of Helsinki (2006-present). Initially, it was geared towards medical surveillance. As opposed to NEXUS, the system consists of three independent extraction tools for three languages (English, French and Russian). While English and Russian components are quite similar and rely on scenario-specific ontologies, gazetteers and patterns [39], the French module uses robust discourse-based extraction rules and gazetteers of location and disease names [29]. Most effort was put into the English module, and the Russian one has been under intensive development only in the recent years. As mentioned by [14], PULS uses an architecture similar to GATE[54] in many respects. It constitutes a multistage pipeline including tokenization, punctuation processing, lexical lookup, morphological analysis, shallow syntactic parsing, NER, anaphora resolution (to handle full articles), pattern matching, inference, template filling, normalisation, and output template generation, where each level processing builds on the results of the previous stage.

As PULS looks at each sentence as a potential event, a relevance score is needed to filter out the unrelated data. The relevance is predicted with machine learning algorithms (SVM and Naïve Bayes) that use predefined features, such as (i) relative position of a sentence within text, (ii) difference between the publication and event date, (iii) document length, (iv) lexical features. Also noteworthy, the system distinguishes between single occurrences of outbreaks, periodic events, and unknown (undiagnosed) diseases.

PULS (English and Russian modules) applies standard grammars (cascaded finite-state transducers) and ontology-based dictionaries to fill the slots for EVENT TYPE (disease name or cross-border crime type), EVENT_DATE, LOCATION (town, village, area, etc.), VICTIM_DESCRIPTOR, VICTIM_COUNT, VICTIM_STATUS (infected, sick, dead), VICTIM_TYPE (human, animal, plant) with strings from text (for

[51] https://github.com/ixa-ehu/NAF
[52] http://kyoto-project.eu
[53] Pattern Understanding and Learning System: http://puls.cs.helsinki.fi/static/index.html
[54] General Architecture for Text Engineering: http://gate.ac.uk

disease outbreaks monitoring). Also, the system adds DOCUMENT_NUMBER, NORMALIZED_DATE, PARENT_INCIDENT and INCLUSION_TYPE to the description. PARENT_INCIDENT and INCLUSION_TYPE data serve to reconstruct previous outbreaks (more details are given in [18]). For cross-border crime monitoring, it populates the slots with DATE, LOCATION, ITEM (smuggled), VICTIM_DESCRIPTOR, PERPETRATOR_DESCRIPTOR. The Epidemics ontology contains around 4000 concepts, and the Security ontology - 1190 concepts (more data is given in [39][14]).

PULS patterns include two elements: verb/object or verb/subject. In the Russian module, patterns match five syntactic constructions. Morphological and syntactic analysis is performed using the AOT[55]. An event is triggered, when a pattern is associated with a concept in the domain ontology. The fact that Russian is a highly inflected and free word-order language constitutes the major difficulty for pattern construction [12].

Xlike project[56] (2012-2014) produces event-related knowledge from news and blogs in English, Spanish, Catalan, Chinese, German, and Slovenian using Newsfeed[57] service and free-share dmoz taxonomy[58]. The main categories of the dmoz taxonomy are shown in Fig. 2.6. This taxonomy is too large, therefore, Xlike developers only make use of up to three category levels (around 1000 categories).

Xlike uses keyword-based clustering to associate similar articles and then performs cross-lingual document linking by means of LSI (Latent Semantic Indexing) and a generalized version of CCA (Canonical Correlation Analysis) on the basis of an aligned multilingual corpus. This technique is used in cross-lingual plagiarism detection [10]. In Xlike system cross-lingual cluster matching precedes the storage into the event registry as opposed to the systems where scenario template comparison is performed after database population (e.g., NEXUS).

The main pipeline of the system is as follows: (i) pre-processing stage (article semantic annotation, extraction of date references, cross-lingual article matching, duplicates removal); (ii) event formation (article clustering, cross-lingual cluster matching, event slots extraction, related events identification); (iii) event storage to the event registry.

Only the most general event data is identified on the cluster level, namely: the title and first paragraph of the centroid article, date (most frequent date mention in an event cluster), location (GeoNames), dmoz category of event cluster, entities and keywords (frequently occurring concepts). Xlike distinguishes between entities and keywords using Wikipedia relations data from infoboxes. Related events are retrieved by measuring

[55] Automatic Processing of Text project tools:http://aot.ru
[56] http://xlike.org
[57] http://newsfeed.ijs.si
[58] http://dmoz.org

FIGURE 2.6: An excerpt from the dmoz taxonomy

```
<Topic r:id="">
    <catid>1</catid>
    <d:Title></d:Title>
    <lastUpdate>2010-03-24 01:16:43</lastUpdate>
    <d:Description></d:Description>
    <narrow r:resource="Top"></narrow>
    <narrow r:resource="Kids_and_Teens"></narrow>
    <narrow r:resource="Bookmarks"></narrow>
    <narrow r:resource="Test"></narrow>
    <narrow r:resource="Adult"></narrow>
  </Topic>
  <Topic r:id="Top">
    <catid>2</catid>
    <d:Title>Top</d:Title>
    <lastUpdate>2010-02-16 08:43:34</lastUpdate>
    <d:Description></d:Description>
    <narrow r:resource="Top/Shopping"></narrow>
    <narrow r:resource="Top/Society"></narrow>
    <narrow r:resource="Top/News"></narrow>
    <narrow r:resource="Top/Science"></narrow>
    <narrow r:resource="Top/Business"></narrow>
    <narrow r:resource="Top/Health"></narrow>
    <narrow r:resource="Top/AOL"></narrow>
    <narrow r:resource="Top/Computers"></narrow>
    <narrow r:resource="Top/Home"></narrow>
    <narrow r:resource="Top/Sports"></narrow>
    <narrow r:resource="Top/Arts"></narrow>
    <narrow r:resource="Top/Regional"></narrow>
    <narrow r:resource="Top/Netscape"></narrow>
    <narrow r:resource="Top/Reference"></narrow>
    <narrow r:resource="Top/Recreation"></narrow>
    <narrow r:resource="Top/Games"></narrow>
    <narrow r:resource="Top/World"></narrow>
  </Topic>
```

concept similarity between events. Each event is represented as a bag of words with TF-IDF weighting, and the similarity between vectors is calculated using the cosine measure [27].

Deep language-dependent linguistic processing up to semantic role labelling is performed on the pre-processing stage (article semantic annotation). The shallow analysis for each language includes language identification, sentence splitting, tokenization, lemmatization, PoS and/or MSD (morphosyntactic description) tagging, NER, conversion from conll to xml. For English, Spanish, and Catalan processing, Freeling system is being used [**?**]. German module uses Apache OpenNLP (tokenization, sentence boundaries detection), Stanford PoS tagger, TreeTagger lemmatization, and Stanford NER. Slovenian uses a lemmatizer and MSD tagger acquired via supervised learning, and conditional random fields-based NER (Mallet toolkit implementation). The model is learned from a corpus of 2173 labeled sentences. Chinese model consists of a tokenizer, a PoS

tagger, and a NER (person, location, and organization) that are constructed using the Hidden Markov Models algorithm [52].

The deep processing is based on the learned models for syntactic parsing, semantic role labeling, and frame extraction. Syntactic parsers are learned using the algorithms of dependency parsing libraries (e.g., Treeler[59]) and the existing dependency treebanks. For some languages, training resources are very limited, e.g. the Slovenian treebank contains only 10 dependency relations. Semantic Role Labeling also builds on Treeler-learned models. The classifiers of the first type are aimed at distinguishing between predicates and non-predicates, and the classifiers of the second type try to predict argument candidates and assign the corresponding semantic roles. This task could not be accomplished for Slovenian due to the absence of a treebank of labeled semantic roles. The conceptual workflow of the semantic frame constructor is as follows: each argument (semantic role) of a predicate is mapped (with VerbIndex[60]) to a participant of the frame using WordNet predicate senses and VerbNet frames. The deliverables provide very few data on each component's performance testing. Currently, the system is functioning as an online service for multilingual dmoz event monitoring (http://eventregistry.org).

ZENON (2001-2011) was developed to assist experts in intelligence reports analysis. It is aimed to cover conflicts (between ethnic groups, political parties), infrastructure problems and person-related events (marriage, meeting, etc.) in the operational area of the German Armed Forces in Afghanistan. The system processes English, Dari and Tajik texts from the KFOR Corpus [23]. In the absence of training sets for Dari and Tajik, in-house language-dependent grammars for mophological analysis, PoS tagging, and verb phrase chunking, as well as gazetteers for NER were developed and integrated into the main GATE pipeline.

The system uses the following workflow. The corpus of HUMINT reports (KFOR) is taken as input for the GATE pipeline, which produces the annotation of noun and verb phrases. The action type is considered to be defined by the verb phrase. Once the action type is identified, the rest of semantic roles is annotated on the basis of FrameNet[61] semantic frames. However, this functionality is not covered by Tajik and Dari modules, because of the absence of the corresponding treebanks. The frames are reported to be filled with NER and PoS tagger results. The reports are reported to be merged using a HUMINT ontology, however, no descriptions of the ontology are provided in the literature. The articles mention only some of the action classes, such as KILL, REPORT, KNOW, COMMAND, PROPOSE, EXPLODE. Authors do not provide any

[59] http://treeler.lsi.upc.edu
[60] http://verbs.colorado.edu/verb-index
[61] http://framenet.icsi.berkeley.edu/fndrupal/

specification of the exact fusion procedure. The data coming from different reports is extracted, combined and presented using XSLT[62] and IEPS[63].

To summarize, key features of the multilingual event extraction systems architecture are presented in the Tables 2.4, 2.5, 2.6 below.

TABLE 2.4: Knowledge bases for geocoding and event predicate detection and linking

Project/Tool	**NEXUS**	**NewsReader**	**PULS**	**Xlike**	**ZENON**
Ontology	In-house concept hierarchy	KYOTO-DOLCE	In-house	Dmoz	In-house HUMINT ontology
Dictionaries	In-house gazetteers	N/A	In-house	N/A	GATE gazetteers
Geo-tagging	GeoNames, multilingual gazetteer, around 600 000 entries	GeoNames, GoogleMaps	GeoNames & own gazetteers	GeoNames	GATE default and handcrafted gazetteers
Other knowledge bases	-	WordNet, SemLink	-	WordNet, VerbNet	FrameNet

TABLE 2.5: Software for data collection, linguistic analysis, and pattern generation

Project/Tool	**NEXUS**	**NewsReader**	**PULS**	**Xlike**	**ZENON**
Data Acquisition	EMM	LexisNexis	EMM	Newsfeed	-
Linguistic Pre-processing	In-house tool CORLEONE	IXA, Alpino	AOT, Proteus	Freeling, Apache OpenNLP, Stanford NLP, TreeTagger, Mallet toolkit, Treeler, etc.	GATE
Pattern Engine	In-house tool EXPRESS	NAF	In-house (Proteus-based)	-	JAPE

TABLE 2.6: Core event extraction approach. Report population and merging algorithms

Project/Tool	**NEXUS**	**NewsReader**	**PULS**	**Xlike**	**ZENON**
Pattern Learning	Weakly supervised bootstrapping	-	Semi-supervised bootstrapping (English), manual (Russian)	-	Manual
Event type detection	Clustering, pattern matching	Ontology-based predicate detection	Pattern matching	Clustering[64], ontology-based predicate detection	Verb phrase annotation
Slot filling	Pattern matching	Semantic frame mapping	Pattern matching	Semantic frame mapping	Semantic frame mapping
Cross-lingual Report Merging	MT of reports and duplicates removal: semantic role disambiguation, victim counting, event type classification	event corefence resolution (supervised pairwise classification)	MT of reports and duplicates removal	LSI, CCA	HUMINT ontology

[64]Hierarchical bisecting k-means

[62]Extensible Stylesheet Language Transformation: http://www.w3schools.com/xsl/
[63]Information Extraction Presentation System. See [15]

2.4 Summary

The considered event extraction systems rely on hand-crafted (and bootstrapped) patterns and knowledge bases (ontologies, predicate databases, dictionaries, gazetteers etc.). The clear tendency is towards models enrichment with knowledge, learning abstract text representation models, based on semantic databases, rather than unsupervised processing. Commonly there are four main stages of event extraction in such systems: (1) data crawling and storage; (2) pre-processing (formatting, clustering, and/or linguistic preprocessing from tokenization to NER); (3) event (anchor or trigger and its arguments) scenario template creation using three main strategies that rely on ontology-based dictionaries: (i) finite state transducer-driven, (ii) semantic frame-driven, and (iii) relation extraction-driven; (4) event merging and storage.

Section 2.2 overviews the approaches used for the general political event detection (2.2.1, 2.2.2), and protest event detection as a more specific issue (2.2.3). Relevant news in the considered systems are usually retrieved using keyword search and learning models (e.g., SVMs, Naïve Bayes). The coding procedure employs a pattern- and dictionary-based (PETRARCH) or probabilistic (Serif, Alex Hanna's project) approaches on sentence and text level. From the perspective of social protest research, the use of the SSP ontology is more fruitful, because it is tuned to civil unrest, the interactions between the authorities and society, in contrast to CAMEO. The main disadvantages of the systems can be roughly outlined as follows:

- inability to process non-English text (unless translated);
- need for large human-labeled datasets for classification;
- limited protest event descriptions by the coders (commonly, we obtain data on source, target, event trigger, date, and location) and ontologies (only four protest event types in CAMEO);
- CAMEO's event intensity is measured from the predefined Goldstein scale[65], initially developed for WEIS, and not from the text itself.

Section 2.3 describes multilingual event extraction approaches and systems. Multilingual news monitoring systems are multilayer and highly modular, some of them are able to perform not only event extraction, but also intelligent story tracking and event relationship analysis. On each processing layer, features are accumulated and further used by pattern matching and frame construction mechanisms. NEXUS and PULS are complementary event extraction engines, however, they differ in language coverage. PULS

[65] http://web.pdx.edu/ kinsella/jgscale.html

is able to analyse cross-border activity only in English and Russian. NEXUS is developing more abstract patterns in an attempt to handle complex verb phrase structure, nested constructions, and other phenomena, which can be encountered in the lead news sentences. The reported performance of this system for cross-border crime and illegal migration is worse than for crisis and violent events. Xlike attempts to populate a global event registry. The main disadvantage of the system is the ontology used, which is basically a collection of topically categorized websites, and does not serve for event classification task. Also, the produced registry entry is a very general event description.

As opposed to political event data-focused systems, multilingual news monitoring systems produce detailed summaries of events, and perform high-quality event merging on the basis of report data, which excludes duplicates. The main disadvantages of multilingual systems can be roughly outlined as follows:

- highly dependent on multiple external processing tools;
- treebanks for under-resourced languages are not enough representative or available;
- insufficient coverage of socio-political events;
- are not publicly accessible;
- not customizable.

Within the scope of this thesis, experiments for testing the application of generic multilingual pipeline for protest event data collection and coding have been conducted. Generally, coding implies event annotation/extraction and code assignment using the codes established in a codebook. Here, we focus on the first stage, namely, ontology-based event annotation and extraction of protest-specific features. News headlines are collected and formatted using Python scripts and further processed within the GATE framework. The output comes in CSV format. Also, multilingual ontology and gazetteer population with slot instances have been carried out. The advantages of our algorithms are as follows:

- sentence-level protest detection;
- event weight determination (size, duration, violence involvement, regular character, intensity of a protest);
- protest event ontology application;
- patterns are simple, generic, and multilingual;
- good runtime performance.

Protest event ontology prototype was constructed independepently, however, it was further refined with regard to the SPP. We opted for the GATE framework, because (i) it is free-source; (ii) its architecture is transparent and customizable; (iii) it provides a relatively fast processing of simple cascaded grammars; (iv) it combines low-level and top-level text processing functionality; (v) allows external tools to be easily adapted to GATE processing. In this implementation, we focus on news headlines, therefore, we do not need to capture the information scattered across sentences. The current version does not include event merging and database storage, because we do not aim to build a fully-fledged system from scratch, but to construct separate modules with multilingual patterns for further integration.

Chapter 3

Multilingual Pipeline Construction

The scope of the EuroPEA (Protest Event Analysis for European languages) pipeline developed within the framework of the present thesis lies in the annotation and extraction of event features that are characteristic of protest events, such as the triple of event type (protest, demonstration, strike, piquet, etc.), location (geographical location where a given event takes place), and reason (protesters' position on an issue), as well as the Protest Weight (includes size, duration, iteration, intensity and violence involved). In this implementation, we elaborate rules for sentence-based event detection, and study the properties of features' mentions by manually examining a representative body of multilingual news headlines (around 5000). Natural language representations of Event_Type, Event_Location, and Event_Reason are linked via an in-house ontology. For the event weight no mappings are currently made, the system provides primary detection and extraction so that large protests and events with increasing scale can be further selected. The date extraction task is left out of the scope of the present work, because it is considered a solved problem in the field. No particular algorithm has been used to detect actors: partial annotations are made using the DBpedia ontology (ethnic, professional, religious, etc. groups). Multilingual generic patterns can be readily used to populate plain gazetteer lists with actor instances, however, currently there is no connection to ontology classes. Basically, to provide actor labeling for protest events, we need a high-quality annotation of animated plural nouns by part-of-speech taggers for each of the languages to detect person groups, and automatically acquired lists of source and target entities, which is a well-known and not particularly challenging task in the domain, but it is too time-consuming, which is why it has been omitted. In this implementation, eight protest event features have been selected, and our task is to correctly detect their boundaries and extract them using experimental generic patterns.

The pipeline includes two main processing stages: event data selection and coding. Event selection implies (i) data acquisition from various multilingual sources with a Python-based crawler, storage in JSON format, and (ii) primary pre-processing of text units (filtering, formatting, date delimitation). Event coding is the process of event data annotation and code assignment. In the present implementation, our ontology does not assign any specific codes, only the English names of the features. The coding procedure includes two substages, namely: linguistic processing and output generation. Linguistic processing includes (iii) Part-of-Speech tagging with external taggers (currently, only Swedish), (iv) tagger results customization with a Python script, and (v) core processing with the Java-based GATE framework: tokenizing, sentence splitting, remote repository lookup, generic tagging (TreeTagger for Bulgarian, French, Polish, Russian, and Spanish), euroPEA ontology and gazetteer lookups, ANNIE gazetteer lookup, grammar, ontology population module, exporter to CSV. The main stages are depicted in Fig. 3.1.

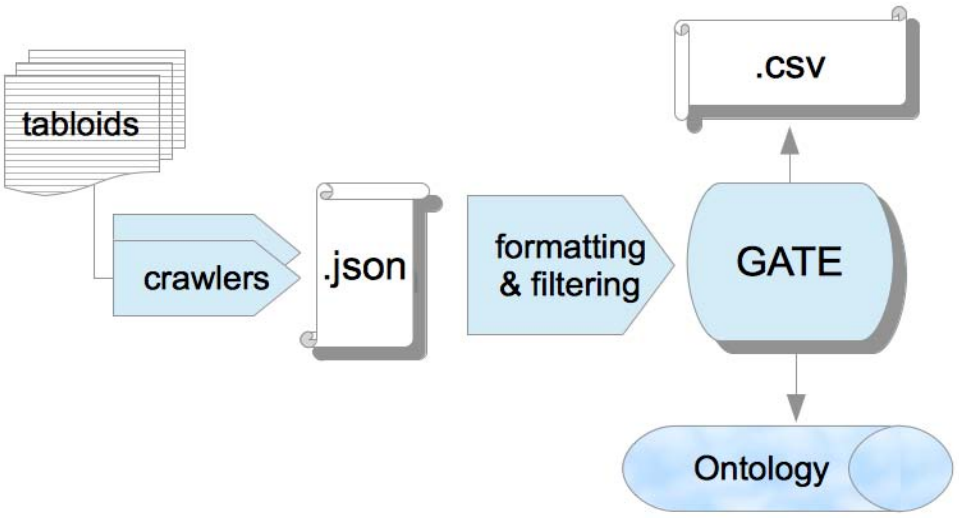

FIGURE 3.1: euroPEA general workflow

This Chapter provides details on mechanisms, order and parameter selection for the pipeline components.

3.1 Acquiring data for Processing

3.1.1 Crawling

The multilingual corpus for our experimental study has been collected using the crawlers developed for each of the languages within the Scrapy web crawling framework[1]. A sample generic code with comments is presented in Appendix C. The scrapers extract news headlines and, optionally, URL, subtitle, metadata, text body, date/time and source. A sample entry of the output file is shown in Fig. 3.2. An article is selected and stored in case of the mutual presence in a given headline of several key phrases from two predefined lists. The first list includes protest TYPE names. The English equivalents are as follows: *demonstration, manifestation, protest, rally, action, boycott, strike, picketing, hunger strike, gathering, parade, procession, march, riot, revolt, civil disorder, civil unrest, rebellion, uprising, mutiny, insurgency*, and *symbolic acts*. The second list contains triggers for the co-occurring concepts, such as LOCATION (prepositions) and REASON (complex prepositions, such as "in support of", "in defence of", etc.). For the purposes of the experiments, a multilingual corpus has been collected using the settings indicated in the Appendix D, where for each of the languages we specify the crawling domains, triggering terms and co-occurring concepts. The crawlers are run via the scrapy crawl `scraper_name > output.json` command.

URL	Title	Subtitle	Metadata	Text_body	Time
http://www.reuters.com/article/2014/08/22/us-yemen-protests-idUSKBN0GM12C20140822	Tens of thousands of Yemeni Houthis protest against govt in capital		Saudi Arabia, Yemen, Abd-Rabbu Mansour Hadi, Ali Abdullah Saleh, Mohammed Al-Sayaghi	Tens of thousands of Yemenis massed in the capital Sanaa on Friday in a protest called by the Shiite Houthi movement, which wants the government to reverse a decision on cutting fuel subsidies and resign...	Fri Aug 22, 2014 4:39pm

FIGURE 3.2: An entry of output.json displayed in a table

3.1.2 Filtering the output

The output of the crawlers is pre-processed as follows. CrawlFilter.py module contains 3 phases. The first phase performs filtering of headlines (and the respective entries) that contain stopwords from the following lists:

```
Latin-character_substrings = [" su marcha a", " puesta en marcha", "protestant",
" en marcha el ", " en marcha la ", "pone en marcha", "poner en marcha", "puso
en marcha", "puesto en marcha", "pondrá en marcha", "ponga en marcha",
```

[1] http://scrapy.org

```
"marcharse", "marchamo", " se marcha ", "Marchand", "Marchamalo", "marcha
atrás", "marchar de", "Todos los contenidos sobre", "RTVE :: buscador:",
"campo de concentración", "marcheur", "à marche forcée", "en marche",
"frammarsch", "démarche", "de marche révolutionnaire", "marche arrière",
"protestanc", "-Zdrowie w Dziennik.pl", "- Auta, Samochody, Motocykle -
Serwis Motoryzacyjny", "wypadek", "wypadk" "Wypadek", "samochod", "pi␣lka
nożna", "- Odpowiedz - Forum"]

Cyrillic-character_substrings =["турмаршрут", "путешестви", "пътешестви",
"нашестви", "Нашестви", "пожар", "оферта", "прибыль", "торг", "откат",
"рынк", "курс", "фонд", "фьючерс", "протестант", "пакет","валют", "покупк",
"покуп", "акционер", "продаж",  "маршрутк", "ДТП", "\%", "Рынок", "акциз",
"рыноч", "экономи", "роисшестви", "маршрутизатор", "вакцин"]
```

These lists have been constructed manually from the most frequent errors of the algorithm. Stopwords include substrings of terms that are characteristic of the topics "traffic accidents", "cars", "market", "religion", "health", phrases like "All the articles about ...", and collocations with the main query keywords ("protest", "march", "blockade") that are reliable indicators of misselection.

In this implementation, we focus on the ontology-based protest event data annotation and extraction, and, therefore, less attention has been paid to the automated refinement of data selection. The above stopwords filter worked for the present collection: the results are described in Chapter 5. The use of machine learning algorithms will be a more robust solution for this case, however, currently we do not have any labeled datasets at our disposal to perform training.

Apart from the stopwords lists application, Phase I does the following:

- removes frequent noise substrings from the headlines: "Protestdemonstration i Kuwait | Utrikes | SvD" (everything that goes after the first vertical bar);
- removes primary total duplicate headlines and the respective entries using the set() function;

Phase II removes partial duplicates and the respective entries using the difflib Python package and NLTK toolkit[2]. difflib.SequenceMatcher compares two string sequences by recursively identifying the longest contiguous blocks free of "junk" values, and ratio() function calculates the ratio. The strings are considered highly similar, if their

[2]http://www.nltk.org/

similarity exceeds the experimentally established threshold (0.85). The source string remains, while the almost-matching strings are removed from the dataset. The second algorithm provided by the NLTK package calculates minimum edit distance between two strings. Minimum edit distance or the Damerau-Levenshtein distance is the number of characters that should be removed, replaced or inserted to a string s_1 to transform it into s_2. A word "rain" can be transformed into "shine" in at least three steps: for instance, "rain" > "sain" (substitution) > "shin" (substitution) > "shine" (insertion). Here, each operation costs 1, and minimum edit distance equals 3[3]. Mathematically, Levenshtein distance is expressed as shown in Fig. 3.3. [4]. Here, $[a_i! = b_j]$ equals 1 if the corresponding characters are not the same. In the minimum, the first element corresponds to deletion, the second one to insertion, and the third to substitution, if the characters mismatch.

Minimum edit distance is slower than SequenceMatcher, however, it captures more sophisticated partial duplicates: for instance, if the same words are mentioned in their morphological variations.

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} 0 & , i = j = 0 \\ i & , j = 0 \text{ and } i > 0 \\ j & , i = 0 \text{ and } j > 0 \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j) + 1 \\ \mathrm{lev}_{a,b}(i,j-1) + 1 \\ \mathrm{lev}_{a,b}(i-1,j-1) + [a_i \neq b_j] \end{cases} & , \text{ else} \end{cases}$$

FIGURE 3.3: The Damerau-Levenshtein distance

Phase III writes unique entries to an output file. The CrawlFilter.py module is called from the command line by indicating the script name, followed by the source folder and the output file.

The DateDelimiter.py module filters the crawled output by date. It is called from the command line with the following arguments: (1) script name, (2) source document, (3) output document, (4) month: one digit for an exact month or two hyphenated digits ("1-12") for a month interval, (5) year: one digit for an exact year or two hyphenated digits ("2000-2015") for a year interval.

3.2 Part-of-Speech Tagging

In the course of our study, the features produced by free-share part-of-speech (PoS) taggers for Bulgarian, French, Polish, Russian, Spanish, and Swedish have been explored.

[3] http://www.nltk.org/_modules/nltk/metrics/distance.html
[4] taken from https://qkdb.wordpress.com/tag/levenshtein-distance/

We have been looking for a tagger with good language coverage, runtime performance, rich set of features (the main PoS tag, followed by other morphological data on a given token) that can be easily integrated into GATE. As a result, we opted for the TreeTagger[5] plugin for GATE that covers all of the considered languages, except Swedish. Treetagger yields satisfactory results, however, the models would benefit from domain-specific training. A brief description of the taggers (Mystem, Freeling, TreeTagger, Stagger) that were integrated into the pipeline at different stages of our study is set out in the present Section.

3.2.1 Yandex Mystem 2.0 Plugin

Mystem provides morphological analysis of texts in the Russian language. The original package with documentation is provided by Yandex developers [6]. The plugin for GATE is an open source tool created in the course of a PhD project[7]. The plugin is compatible with an earlier version of Mystem (2.0), which does not support homonymy disambiguation and entity recognition. The produced annotations list each of the PoS features independently, and not in a single feature code, as in Freeling and other taggers. For instance, one of the analysis versions for the word "минимальным" (adjective "minimal" in the instrumental case) is as follows:

```
baseForm: "минимальный"
case: instrumental,
form: full,
gender: neuter,
multiplicity: singular,
pos: adjective.
```

In the absence of the context disambiguation option, each word gets annotated with multiple morphological analysis versions. It results in a low runtime performance and high level of ambiguity. For instance, Mystem annotates some prepositions as nouns, and they are wrongly recognized as noun modifiers by the noun phrase chunker. Due to ambiguity issues, additional constraints are added, and patterns become heavier and more error-prone.

Mystem was integrated into our pipeline in the earlier experiments on Russian news and social networks analysis [11] [12]. Rules for the detection of Actor, Location,

[5] http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/
[6] https://tech.yandex.ru/mystem/
[7] https://code.google.com/p/mystem-morphtagger/

FIGURE 3.4: Mystem annotation use in a location detection rule

```
(
//1: Match location prepositions.
({Token.string ==~ "[Нн]а"}|{Token.string ==~ "[Вв]"})

//2: match gazetteer lookup or an upperinitial substantive in the
ablative case and more than 3 characters long (unknown location).
({Lookup.majorType == location}|{Token.orth == upperInitial,
Morph.pos == substantive, Morph.case == ablative, Token.length > 3})
)
```

Type, and Reason in Russian protest-related headlines were elaborated. For instance, the Location grammar includes five rules that take into account the noun case, form, animation, and other features. A sample rule to detect location circumstance (mostly country names) is given in Fig. 3.4.

To sum up, Mystem provides rich morphological features that can enhance the disambiguation of entities in Russian datasets, however, due to its particular tagset mapping (separate features), monolinguality, and the above mentioned issues, it has not been employed to construct the multilingual generic patterns.

3.2.2 Freeling 3.1

Freeling library [34] provides linguistic analysis support for around 10 languages. It covers Spanish, French and Russian from our selection. Freeling library contains 76.000 lemma-PoS combinations for Spanish, 54.000 lemma-PoS combinations for French, and 510,000 lemma-PoS combinations for Russian. There are two part-of-speech tagger modules available: a trigram Markovian tagger and a system integrating statistical and rule-based approach (relax_tagger). In the course of our experiments, the output of the Hidden Markov Models-based tagger (the first module) has been converted into a GATE xml via a C++ script (Appendix E). Probabilities of unknown words are generated on the basis of their endings. The resulting feature set includes lemma, PoS code, and probability. PoS codes are established in accordance with the EAGLES tagset[8]. Each code comprises a part of speech feature accompanied by other morphological features of a given word. For the word "come" (Spanish, "eats"), the analysis is as follows:

```
lemma = comer,
pos = VMIP3S0,
```

[8] http://www.ilc.cnr.it/EAGLES96/annotate/annotate.html

```
probability = 0.75.
```

(V - verb, M - principal, I - indicative, P - present tense, 3 - third person, S - singular, 0 - zero value if the feature cannot be identified for this form)

A number of difficulties arises when adapting the program output to be processed by GATE, due to the required conversion of the input text to ISO-8859 encoding family. The text has to be encoded to ISO and than formatted back to UTF-8, which often becomes problematic, because of undecodable characters e.g. in articles coming from social networks. Also, Freeling fails to recognize future tense forms of the verbs like "to protest", "to rally" in the Russian language, which is important for the determination of the event status.

All in all, Freeling provides high-quality PoS tagging for three languages of our selection on the basis of the EAGLES tagset standard, however, alone it does not satisfy the needs of our system and requires additional pre-processing. We conclude that it can be integrated at a later stage of the system development for disambiguation purposes, if no other solution comes up.

3.2.3 The Stockholm Tagger of Swedish

The Stockholm tagger (Stagger) is one of the best-performing taggers for the Swedish language with the per-token accuracy of about 96.6 percent. It is based on Collins averaged perceptron [57] and it is implemented in Java. The output is made in the plain format with one token/tag pair per line ("`token<space(s)>tag`"), where tag features are separated with vertical bars. Stagger uses the Stockholm-Umeå Corpus tagset. A sample output for the Swedish sentence "Protest mot minskat stöd till kulturstidsskrifter" ("Protest against the reduction of cultural journals support") with tag details is given in Fig. 3.5.

For the purposes of our system, the PoS tagger output is converted to GATE XML using a special Python script (Appendix F).

3.2.4 TreeTagger Plugin

The TreeTagger [42] is a highly multilingual PoS tagging package that is maintained and updated by a large community of contributors. In 1995, it was reported to solve the problem of trigram taggers by using decision trees in the estimation of transition probabilities from sparse data. In terms of our requirements, the TreeTagger is preferable due to the following reasons:

FIGURE 3.5: A sample Stagger output

```
Protest NN|UTR|SIN|IND|NOM //comment: Noun|Common(gender)|Singular|Indefinite|Nominative

mot PP
//comment: Preposition

minskat PC|PRF|NEU|SIN|IND|NOM
//comment: Participle|Perfect|Neutral(gender)|Singular|Indefinite|Nominative

stöd NN|NEU|SIN|IND|NOM
//comment: Noun|Neutral(gender)|Singular|Indicative|Nominative

till        PP //comment: Preposition

kulturtidsskrifter        NN|UTR|PLU|IND|NOM
//comment: Noun|Common(gender)|Plural|Indicative|Nominative

.        MAD
//comment: major delimiter
```

Language coverage. TreeTagger provides lemma and morphological data annotation for Bulgarian, French, Polish, Russian, Spanish.

Compatibility with GATE. TreeTagger has a ready wrapper for GATE. We only have to create parameter files and fill them with the corresponding paths.

Runtime performance. TreeTagger annotates 500 sentences in around 2.8 seconds on our hardware (Macintosh platform, 4 GB, 1.8 GHz Intel Core i5). As a plugin, it needs no formatting, which also saves time.

Morphological features. Language-specific contributions are made by different authors, however, tagsets are similar and can be handled with regular expressions. For instance, Russian PoS tags for Noun, Adjective, and Verb can be presented as follows (hyphen denotes zero feature value):

```
N[cp-][fnmc-][ps-][ngdailv-][yn-]
A[f-][pcs-][fmn-][sp-][ngdail-][fs-]
V[m-][imgp-][fps-][123-][sp-][fmn-][amp-][fs-][ep-][ngdail- ]
```

Bulgarian, Polish and Russian TreeTaggers enrich annotations with sets of morphological features similar to the ones above, while Spanish and French taggers provide little morphological data (almost bare PoS tags). PoS tagging builds on the token annotation, which recognizes compound nouns with a hyphen as one token. It is favorable, because other taggers (Mystem, Freeling) tend to produce several annotations instead of one or simply leave these sequences as unknown.

A sample pattern that matches finite verb forms accross the considered languages is as follows:

```
(
  {Token.category ==~ "VLfin|(Vp[ip][it]f)"}|
  {Token.category ==~ "praet|fin"}|
  {Token.category ==~ "Vmi[spf]"}
)
```

A sample pattern that matches noun forms accross the considered languages is as follows:

```
(
{Token.category =~ "^N|subst"}
)
```

The regular expression operators "= " and "== " in a JAPE grammar[9] match a substring and the whole string respectively. Vertical bar is a logical disjunction operator. Morphological information is assigned to the category feature of the Token annotation. "ˆN" matches the character "N" at the beginning of the string.

TreeTagger yields satisfactory results within the current implementation, however, there is a number of significant drawbacks that can be attributed to the insufficient training:

- the Spanish tagger does not distinguish between Indicative and Subjunctive mood;
- the Bulgarian tagger does not identify infinitive forms;
- in slightly unusual contexts, label assignment of the main clause constituents fails, as in "Una veintena de afectados por posibles desahucios protesta en Alicante" ("Around twenty people affected by potential housing evictions protest in Alicante "), where "protesta", the main verb, gets wrongly labeled with "NC" (common noun);
- the taggers tend to annotate unknown words or entities (capitalized nouns, abbreviations) as verbs: e.g., in Bulgarian, "Шарли" (Charlie Hebdo) is assigned "Vpitcam-p-i", "Орбан" (Hungarian prime minister Viktor Orban) - "Vppicao-p-d", in Spanish, "ERE" ("Expediente de Regulación de Empleo" or layoff) - "VLfin",

[9] http://en.wikipedia.org/wiki/JAPE_(linguistics)

"Ahmadineyad" (Mahmoud Ahmadinejad, an iranian president) - "VLfin", "okupa" ("squatter") - "VLfin", etc.;

- TreeTagger, together with the other presented taggers, does not recognize substantivized parts of speech.

These drawbacks are the main source of euroPEA's inaccuracies. Some of the solutions that can be undertaken to minimize them in the next implementation are as follows: (i) integration of PoS data coming from high-quality external taggers, or (ii) retraining of TreeTagger modules.

TreeTagger is launched via the Generic Tagger PR[10]. The default runtime parameters are left unchanged, except the tagger binary file (the corresponding parameter file should be uploaded) and encoding (should be set to UTF-8). Also, an additional JAPE grammar from the TreeTagger plugin package is added to fix the incorrectly identified lemmas for further gazetteer processing.

3.3 Main GATE Pipeline

euroPEA.gapp is a main corpus pipeline that uses a selection of the subsequent processing resources (PR) in the following order:

Pre-processing stage:

- Document Reset PR: removes annotation sets with their contents created in a previous session;
- Document Normalizer: removes smart quotes, if any, from the document content;
- GATE Unicode Tokeniser: a customizable tokenizer, which by default distinguishes between the following token kinds: word (one token or two hyphenated tokens), number (any sequence of digits), symbol (currency or other), punctuation (each punctuation mark is a separate token, except for hyphen between two tokens without spaces), and SpaceToken as an independent annotation type. Word tokens have an orthografic representation attribute "orth" that provides annotations for tokens with initial capitals (upperInitial), uppercase (allCaps), lower case (lowerCase), or mixed letters (mixedCaps)). Token annotations are supplied to the PoS tagger for further processing.

[10]Tagger Framework: https://gate.ac.uk/sale/tao/splitch23.html#sec:parsers:taggerframework

- ANNIE Sentence Splitter: a cascaded grammar that produces two types of annotations that are used by PoS tagger: Sentence (sentence contents) and Split (sentence marking stops). Split commonly differentiates between internal combinations of dots, exclamation and question marks ("Split.kind == internal"), and external marks that occur before the newline ("Split.kind == external"). The original ANNIE Sentence Splitter rules have been modified to omit internal splits in the headlines. An annotation of type Sentence commonly covers the headline up to the external split, and its offset and ID are used to create Protest_Event annotations.

Part-of-speech tagging stage:

- Generic Tagger PR: a generic tagger that allows using external taggers from Tagger Framework (in our case, TreeTagger). The encoding parameter is set to UTF-8.
- fix-treetagger-lemma.jape: a grammar piped together with the generic tagger to improve lemma annotations.

Gazetteer lookup stage:

- Large KB Gazetteer PR: a tool (Large KB Gazetteer PR) that supports connection to remote repositories via SPARQL queries to endpoints and produces the corresponding ontology-aware annotations with class and instance URI as features.
- AuxGazetteer: a standard ANNIE gazetteer connected to our lists of auxiliary terms (prepositions, numbers, etc.). Runtime parameters: caseSensitive is set to false.
- EThashonto: in-house ontology-aware hash gazetteer (OntoGazetteer PR) of event type. Runtime parameters: caseSensitive == false.
- ETflex: a feature-aware gazetteer (Flexible Gazetteer PR) that matches Token.lemma annotations against EThashonto entries and allows capturing words regardless of their morphological variations.
- ETbwponto: in-house ontology-aware approximate gazetteer (BWPGazetteer PR) of event type. Runtime parameters: caseSensitive == false, wholeWordsOnly == false, normalizedDistanceThreshold == 0.15.
- ERhashonto: in-house ontology-aware hash gazetteer (OntoGazetteer PR) of event reason. Runtime parameters: caseSensitive == false.

- ERflex: a feature-aware gazetteer (Flexible Gazetteer PR) that matches Token.lemma annotations against ERhashonto entries.
- ERbwponto: in-house ontology-aware approximate gazetteer (BWPGazetteer PR) of event type. Runtime parameters: caseSensitive == false, wholeWordsOnly == true, normalizedDistanceThreshold == 0.15.
- EWhash: a hash gazetteer (Hash Gazetteer PR) of event weight. Runtime parameters: caseSensitive == false.
- EWflex: a feature-aware gazetteer (Flexible Gazetteer PR) that matches Token.lemma annotations against EWhash entries.
- EWbwp: an approximate gazetteer (BWPGazetteer PR) of event weight. Runtime parameters: caseSensitive == false, wholeWordsOnly == false, normalizedDistanceThreshold == 0.15.
- ANNIE: a standard GATE gazetteer used to annotate location and person names lookup. Runtime parameters are set to default: longestMatchOnly == true, wholeWordsOnly == true.
- ELbwp: an approximate in-house gazetteer (BWPGazetteer PR) of event location. Runtime parameters: caseSensitive = true, wholeWordsOnly = true, normalizedDistanceThreshold = 0.15.

Grammar processing stage:

- Jape Plus Extended Grammar PR: an extended version of the new Jape_Plus transducer providing additional constraints, and access to JapeUtils class; the main event detection grammar with default runtime parameters.

Gazetteer population:

- Gazetteer List Collector PR: populates gazetteers with labeled instances of a given type.

CSV export stage:

- Configurable Exporter PR: a generic exporter, the export format is set in example.conf.

PoS tagging procedure has been described in detail in the previous Section 3.2. In the following subsections, we focus on the core components, namely: ontology and gazetteer lookup, event extraction grammar, and application output.

3.3.1 Ontology and Gazetteer Lookup

The order of ontology and gazetteer lookup components is given in the previous Section. First, Large KB Gazetteers independent of other PRs and connected to an online endpoint are processed. The gazetteer folder contains a parameter file (config.ttl), where the endpoint is indicated, and query.txt with the corresponding SPARQL query. The queries are specified in the next Chapter. Second, the lists of auxiliary terms, such as prepositions, conjunctions, negative particles, modal verbs, speech act verbs, date units, months, and numbers, are processed via the standard finite state machine-based ANNIE gazetteer. ANNIE gazetteers have useful runtime parameters, such as longestMatchOnly, wholeWordsOnly to experiment with. In this gazetteer they are set to default. Next, Event_Type, Event_Reason, and Event_Weight lists are piped using the same strategy: an OntoGazetteer (or Hash Gazetteer for Event_Weight) is created, which is connected to the gazetteer lists via lists.def and ontology mappings via mappings.def. The caseSensitive parameter is set to false, the encoding is the default UTF-8, and the Hash Gazetteer is specified by default. In hash gazetteers, the search is performed in HashMaps rather than in Finite State Machines, which is reported to consume less memory and be faster, in case large lists are processed. The output of this hash-based OntoGazetteer is consumed by the Flexible Gazetteer PR that matches the necessary features (in our case, word baseforms in Token.lemma). Next, an OntoGazetteer based on an approximate gazetteer (BWPGazetteer PR) calculates the Levenshtein distance between the list entries and text strings. If the minimum edit distance between a string and a given entry is below the established threshold, the string gets annotated. The experiments with the normalizedDistanceThreshold showed that the optimal value is in most cases the default one (0.15). The use of normalized distances allows to mitigate the problem of varying string length: the relative distance threshold in range [0.0, 1.0] is specified by the user, and the absolute distance threshold for each searched word is computed using the multiplication of the relative threshold by the string length [35]. The WholeWordsOnly parameter is set to false for the Event_Type and Event_Weight lists to capture word forms that have been missed by the tagger, as well as compound words' parts. Next, an ANNIE gazetteer containing person and location names in Latin and Cyrillic scripts is piped (standard packages for English, Russian, and French). Finally, an approximate gazetteer of our location lists is processed to capture morphological variations of the entries.

3.3.2 JAPE Grammar

JAPE or Java Annotation Patterns Engine is a formalism devised for regular expression patterns transduction into annotations. Each grammar is a finite-state automaton.

FIGURE 3.6: A sample JAPE rule

```
Phase: SymbolRemover
Input: Token


Rule: RemoveStuff
(
{Token.string =~ "[_\"'«»%()№“”+|=,,]"}
):token
-->
:token {inputAS.removeAll(tokenAnnots);}
```

Unlike in Turing machine model, the movement of the read/write head in a finite state automaton is restricted to only one direction, as explained in [26]. The input sequence is mapped into the output sequence having passed through a set of rules. A JAPE grammar may include one phase or a cascade of phases, each of which consisting of pattern/action rule pairs. The first lines of each phase determine the phase name (Phase:), input annotations (Input:), and the way patterns are matched (Options: control:). Nextly, macro patterns are introduced, if any, and rules are listed. The order of phases is indicated in the main.jape grammar, which is the one that is uploaded to GATE as Jape or Jape Plus (Extended) PR. The left side (LHS) of each rule (before the symbol "−− >" of transduction) contains a pattern that may use a (ontology-based) gazetteer lookup or other annotations (tokenization, sentence splitting, morphological analysis). The right side (RHS) of each rule (after the symbol "−− >" of transduction) manipulates the produced annotations. The set of JAPE expressions for the RHS is rather limited, therefore, the use of Java code is allowed. A sample rule that removes some of the unnecessary symbols on the basis of Token annotations and uses Java in the RHS is given in Fig. 3.6.

Our main grammar contains 12 main phases (also called grammars) and two additional phases for ontology population (Fig. 3.7). The SymbolRemover phase filters out special symbols and internal splits (sentence end markers, such as full stops, within headlines). ETgrammar is a single-pattern phase that annotates event type (verb, noun or adjective) with the corresponding ontology class. ERgrammar (7 patterns) labels left and right parts of the event reason annotation, which are merged at the next stage in ERmerge phase. ELgrammar (4 patterns) highlights event location, and EWgrammar (I, II, III, IV) annotates event weigth indicators: I - Duration (3 patterns), II - Iteration (3 patterns), III - Size (5 patterns), IV - Intensity and Violence (2 patterns). FILTER refines the indicators of event weight that form part of compound nouns by removing false positives. ToSentence phase collects the features of all previous annotations and moves

FIGURE 3.7: main.jape

```
MultiPhase: euroPEA
Phases:
SymbolRemover
ETgrammar
ERgrammar
ERmerge
ELgrammar
EWgrammarI
EWgrammarII
EWgrammarIII
EWgrammarIV
FILTER
ToSentence
ToMention
ToOntology
REMOVE
```

them to the Sentence annotation (headline). It is further renamed to "Protest_Event" within the same phase, and an event ID is added to the feature map. ToMention assigns ontology URL and class feature to the annotations of a given type. ToOntology collects instances (strings) that belong to a given class (e.g., Issue of protest) and performs ontology population. A detailed description of event feature grammars is given in the next Chapter. REMOVE phase deletes the annotations used to create Protest_Event.

3.3.3 Application output

In our experiments, we have developed several output options for the obtained features: (i) export event features to CSV using the Configurable Exporter PR, (ii) populate ANNIE gazetteers with new entries (string values that should be further classified/clustered), and (iii) populate ontology classes with new class instances to be used together with the ontology outside GATE.

Export to CSV. Protest event features for each of the events are exported to the CSV (comma-separated value) format to facilitate their further processing. An exporter is run for this purpose that is configured via euroPEA.conf file (Configurable Exporter PR). The configuration file contains one line of text, where feature types are specified as follows: `"{Protest_Event.Event_ID}", "{Protest_Event.Event_Type}", "{Protest_Event.Position}", "{Protest_Event.Issue}"` , etc.. output.csv contains one event per line, represented by comma-separated features: `"145270", "protest", "against", "здравен законопроект"` (draft law on healthcare services), etc..

Gazetteer population. ANNIE gazetteer lists of Event_Location, Event_Reason issue and Event_Actor are populated via Gazetteer List Collector PR to facilitate the further annotation of these instances (in particular, new strings, unknown to the current

gazetteers) in new data. Runtime parameters define the annotation types to be collected, the gazetteer to be populated (ANNIE), annotation set name (empty in our case). Within ANNIE, three empty gazetteers with their names corresponding to the specified annotation types are created. A sample populated gazetteer list of Event_Location is presented in Fig. 3.8.

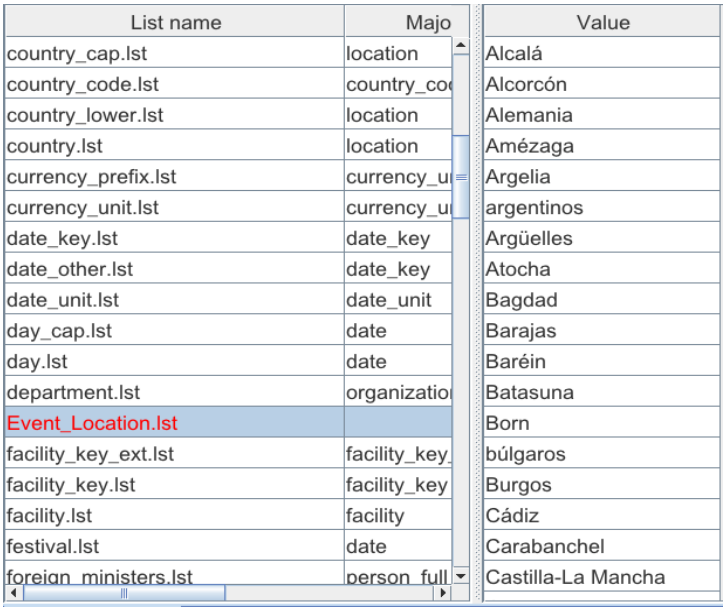

FIGURE 3.8: Event_Location list populated with the produced annotations

Ontology population. In the current implementation, we make use of two ontology population phases: ToMention (creates an annotation of type Mention on the basis of a selected annotation, such as Event_Location or Event_Actor, and enriches it with the corresponding class and ontology URL) and ToOntology (produces instances of a given class within the specified ontology using Mention annotations). The RHS code is based on the official samples provided on the GATE website[11]. The ToMention sample code is shown in Fig. 3.9 that produces labels with the Event_Location class URIs.

An excerpt of the ToOntology code with comments is presented in Fig. 3.10. The operations have the following order: (1) iterate through Mention annotations, (2) create rdfs:label for a given Mention, (3) create Literal value for a given Mention, (4) get the Mention class feature, (5) create URI for the class name, (6) get all the existing instances of the class, (7) iterate through these instances, (8) see if any of the existing instances corresponds to new annotations, (9) if it does, consider the coinciding annotations the same, (10) if not, create an instance with a generated name, (11) label it (add rdfs:label property) with the covered text (Literal value), (12) store the instance URI on the annotation.

[11] https://gate.ac.uk/wiki/TrainingCourseJune2012/track-2/module-6-code.html

FIGURE 3.9: ToMention.jape

```
Phase: ToMention
Input: Event_Location
Options: control = appelt

Rule: CreateMention

({Event_Location}):mention
-->
{
gate.AnnotationSet mentionSet = (gate.AnnotationSet)bindings.get("mention");

FeatureMap features = Factory.newFeatureMap();
features.put("ontology", ontology.getURL());
features.put("class", "Event_Location");

outputAS.add(mentionSet.firstNode(), mentionSet.lastNode(), "Mention", features);
}
```

FIGURE 3.10: An excerpt from the ToOntology phase code

```
Rule: AnnotationToInstance

({Mention}):mention

-->
:mention{

//iterate through mention annotations
Annotation mentionAnn = mentionAnnots.iterator().next();

//create a rdfs:label property for the annotated text
AnnotationProperty rdfsLabel = ontology.getAnnotationProperty
(ontology.create OURI(OConstants.RDFS.LABEL));

//create a Literal for the annotated text
Literal text = new Literal (gate.Utils.stringFor(doc,mentionAnnots));

//get the Class name and create the corresponding URI
String className = (String)mentionAnn.getFeatures().get
(gate.creole.ANNIEConstants.LOOKUP_CLASS_FEATURE_NAME);
OClass aClass = ontology getOClass(ontology.createOURIForName(ClassName));
if(aClass == null){System.err.println("Error class \"" + className + "\" does not exist!");
return;}

...
```

The output of ontology population phases is depicted in Fig. 3.11. The left column lists strings annotated by euroPEA.gapp as Event_Location in a Spanish text and automatically attributed to the Event_Location class.

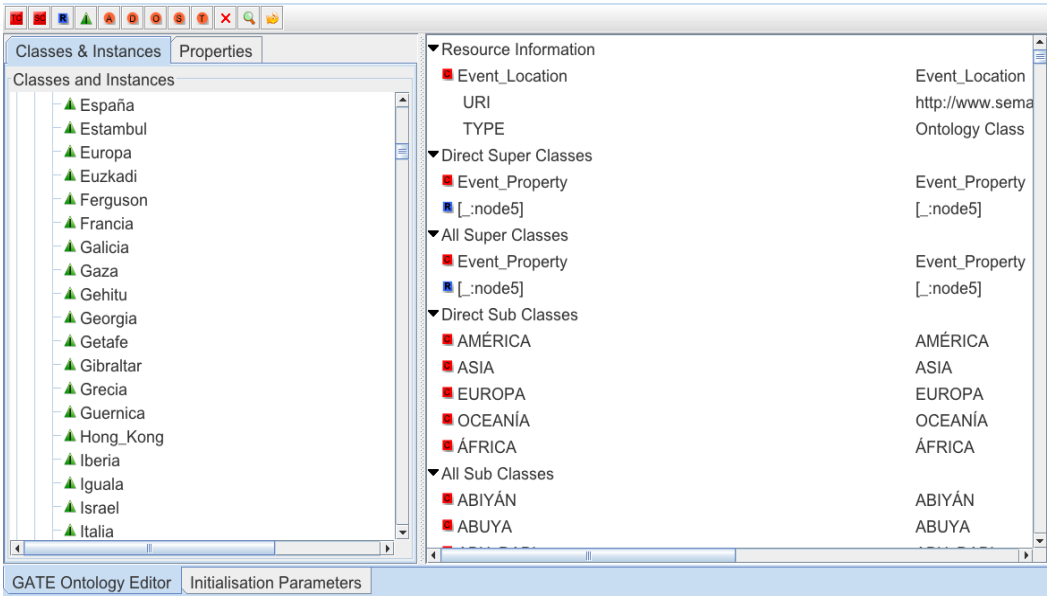

FIGURE 3.11: Ontology populated with Event_Location mentions (GATE screenshot)

The difficulty in the ToOntology code that has been overcome is the creation of a URI-compatible string from a given annotation mention independently of the script (Latin or Cyrillic in our case) for the instance URI part. The string value of the Mention annotation has been converted into a URI part using the OUtils.uriEncode() method.

Chapter 4

euroPEA.gapp: Knowledge Resources

4.1 Domain Ontology

4.1.1 General Approaches to Ontology Construction

An ontology can be constructed in a manual, semi-automatic or automatic way using natural language processing and machine learning. Ontologies can be generated either directly from text or from dictionaries, thesauri, knowledge bases, semi-structured and relational schemata [30]. Learning pipeline includes term extraction, disambiguation, concept identification, concept hierarchy construction, identification of relations and rules within the ontology. Term extraction uses either indexing mechanisms from the information retrieval domain or natural language processing. For the context-based term disambiguation, clustering along with the use of association measures to detect statistically correlated pairs is applied. Thesauri and dictionaries are also employed to group terms with similar meanings. For concept identification, unsupervised machine learning techniques are widely used. In some approaches to concept hierarchy construction, lexical relations of hyponyms are extracted from corpus using automatically acquired context-based lexico-syntactic patterns. Also, an approach based on Harris's Distributional Semantics Hypothesis [21] has been applied. It takes into account the correlation between a word and a context. Contexts are encoded in term vectors, clustering is performed and, finally, a distance measure (TF-IDF or chi-square) is applied to separate term senses. The identification of nonhierarchical relations within an ontology involves the use of text mining techniques and linguistic analysis. The automatic rule identification (a rule looks like "X caused Y", "Y is triggered by X", etc.) is a less developed area,

where there are no common and well-established approaches [53]. The interaction with the user or expert at different stages, as well as the comparison to the existing ontologies and term hierarchies, contribute to the ontology refinement.

4.1.2 EuroPEA Protest Ontology

In this implementation, we manually construct a gold standard ontology of protest events, which is to some extent similar to the SSP, on the basis of the main triggering concepts (event types), and co-occurring concepts, such as preceding events, consequent events, geographical attributes, and other. The core SSP ontology is briefly described in Chapter 2. Our study focuses on the real-time dynamics of the daily protests and their attributes reported in the news of specific countries, therefore, the top categories constitute (1) protest type hierarchy, (2) events preceding protests, (3) events following protests, including authority response, and (4) event properties.

The gold-standard ontology is formalized in Stanford's Protégé-4.3[1] and revised by a domain expert in order to provide quality semantic annotation to the data via the GATE 8.0 framework. It is built using the real data from news feeds. The current version spans event type, reason, actor classification, antecedents, consequences, and event properties. All these data have been considered within the same ontology in order to visualize the dependencies, if any, between events with their attributes, and subevents. The class hierarchy is based on the analysis of around 5000 unique news headlines. SSP classes are taken into account, as well as various resources on the Web, including the interactive access to DBpedia ontology classes and relations[2], in order to select the most relevant features. Firstly, lists of events constituting protest activity, as well as those preceding and following it, have been analyzed and organized into a structure. Secondly, event properties (actors involved, geotemporal data, event impact, status, and reason) have been classified.

Hierarchy. Protest activity is divided into Verbal_Expression and actual Protest _Action. The latter can be a Non-violent_Resistance, and Violent_Attacks. These classes are not disjoint, because such events may anticipate or follow one another. Non-violent_Resistance includes Strike, Hunger_Strike, Business_Strike, March, Picketing, Mass_Demonstration, and other terms. Violent_Attacks include riots, mutinies, rebellions and takeovers. Event_Property class covers event reason, impact, status, actor, location, and date. Event_Reason is an Event_Property that divides actions into the

[1] http://protege.stanford.edu/
[2] gFacet tool: http://visualdataweb.org

expressions of against, inSupportOf, forAndAgainst, toDemand, toConmemorate or relatedTo (for ambiguity cases and sentiment-free cause indications). inSupportOf, toConmemorate, toDemand and relatedTo categories imply a protest against the status quo. The weight of an action is indicated by its Duration (specified in the number of days/weeks/months/years), Iteration (a new event occurs in case the demands are not satisfied), Size (specified in the number of participants or unspecified as "massive"), Intensity (increasing) and Violence_Involved (peaceful and violent) features. The status of the action is Planned/In_Progess/Finished/Never_Took_Place. Actor category includes individuals, unnamed and named groups of people, governing authorities, political parties, church representatives, enterprises, law enforcement, etc., that can be initiators, targets, victims, supporting entities or participants of a protest event. Event_Location classifies country-city data for 5 regions: Africa, America, Asia, Europe, and Oceania. Antecedent_Type includes Appeal (by a party, an individual, etc..), Ban (by an authority), Cancel (by the initiating group or authority), Authorize (by an authority), Warn_on_Disruption (anonymous or authority-initiated disruption threat), Change (a change of event time by authorities or organizers, a change of march route by authorities or organizers), Request (a formal request to the authorities), Threat (a threat of protest by the initiating group) categories, and other. The Aftermath_Type covers the reported Damage, Event, Response, Violence, Property_Capture with the corresponding subcategories. In this Section, we present an excerpt of the ontology as a screenshot (Fig. 4.1). The concept hierarchy (down to three levels, except Event_Type) is shown in the Appendix A.

Event type issues. In attempt to make an experimental classification, we examined and compared the data from online knowledge resources (WordNet, WordReference, Oxford, Wikipedia, Wiktionary, Merriam Webster, etc.), terminology-related discussions by social scientists, popular sociological articles, linguistic studies [13] and previous domain-specific ontologies (SSP). Our aim here is to cover the main types of protest events reported in news stories. The current list includes the following classes with the corresponding subclasses:

I. Non-violent_resistance {comment: non-violent resistance of civil groups against a particular authority}

protest {comment: the act of protesting, a public expression of dissent}

- boycott {comment: a voluntarily ban of a person, organization or country for social or political reasons}
- direct_action [SAME-AS protest_action] {comment: a collective action aimed at satisfying the demands for social or political reasons}

street_protest:

- demonstration [SAME-AS demonstration, manifestation, remonstration, rally, crusade] {comment: a collective action (commonly, massive) consisting in a mass march formation starting or ending with a meeting to hear speakers}
- casserole {comment: a street protest, where people call attention by banging utensils from their homes and in the street}
- concert [SAME-AS musical_protest, concert-rally, concert-protest] {comment: a performance against or in support of the policies of a governing authority}
- flash_mob [SAME-AS flashmob] {comment: a demonstration for political or social reasons that consists in a group of people assembling suddenly in a public place and doing something unusual and seemingly pointless before quickly dispersing}
- escrache {comment: (originating from Spanish-speaking countries) a demonstration by a group of activists held at the doors of authorities' homes or workplaces in order to influence their decisions}
- blockade
  - picket {comment: a non-violent attempt to dissuade people from entering somewhere or "crossing the picket line" by using negative publicity in order to put pressure on a target entity}
  - sit-in {comment: a non-violent protest in the form of the occupation of a location in order to put pressure on a target entity}
- march [SAME-AS procession, parade]
  - torchlight_procession
  - pride_parade [SAME-AS LGBT_parade] {comment: an action in defence of gay rights}
  - silent_protest [SAME-AS silent_parade]

- symbolic_acts {comment: any voluntary actions undertaken as an expression of protest}
  - hunger_strike
  - resignation

- job_action [SAME-AS industrial_action] {comment: collective action aimed at reducing work productivity}
  - work-in {comment: a collective refusal to leave a workplace as a protest against dismissals or closure}

  - business_protest {comment: work stoppage initiated by business owners aimed at improving the conditions of their businesses}
  - slowdown [SAME-AS go-slow] {comment: a job action consisting in employees seeking to reduce their efficiency to put pressure on their management}
    * work-to-rule [SAME-AS rule-book_slowdown] {comment: a job action consisting in people doing the minimum required by their contracts and refusing to do overtime}
  - strike {comment: work stoppage initiated by employees}
    * general_strike {comment: a strike, in which a large percentage of workers of a city, region, or country, takes part}
    * sit-down {comment: a strike, in which employees refuse to leave the premises until their demands are satisfied}
    * sympathy_strike {comment: a strike in support of other employees being on strike}
    * walkout {comment: a strike consisting in leaving the premises and going to the street as an act of protest}

insurgency {comment: a non-belligerent rebellion, a complex of civil resistance measures against the actions of a governing authority}

II. Violent_attacks

- sabotage {comment: a deliberate destructive or obstructive action against a polity or corporation}
- riot [SAME-AS mass_disorders, mass_disturbances] {comment: a spontaneous short-term politically-motivated attack of a property, organization or people by non-governmental actors}
  - race_riot
  - food_riot
- mutiny {comment: an armed uprising (military, marine, or other), spontaneous or organized by a group of people and aiming to overthrow the lawful authorities}
- uprising [SAME-AS rebellion, insurrection, revolt] {comment: massive, spontaneous or organized, violent or non-violent (see: insurgency) actions of non-governmental actors against the position (illegal, discriminative or other) of political leaders}
  - spontaneous

  - organized

- revolution {comment: an organized violent massive action aimed at taking over the governing authority and establishing a new regime}
- Coup_d_État [SAME-AS overthrow, putsch] {comment: an attempt of a particular group to usurp power}

In the course of the actual news reports analysis, we came to a conclusion that, as it is in the colloquial speech, in news reports some of the terms are used as synonyms without the account of distinguishing semantic subcomponents. For instance, an equal use of the words "riot" and "civil disorder", "riot" and "revolt", "riot" and "mutiny", etc., for the same events in the considered languages is observed. Online dictionaries define most protest events basically as "acts of protest". WordReference thesaurus (2015) lists the following interpretations for the term "revolt": uprising, mutiny, revolution, strike, rebellion, which is confusing. The Wikipedia multilingual alignment for the problematic concepts "civil disorder", "riot", "mutiny", and "rebellion" ("revolt") is given in Tab. 4.1. We use the entries of the "Oxford Advanced Learner's Dictionary of Current English" [25] and English Wikipedia articles as starting points to compare the definitions.

In [25] "Civil disorder" is defined as an "angry outburst of rioting caused by political troubles". In the English Wikipedia this term spans obstructive actions, such as illegal parades, sit-ins, riots, whereas the Spanish "desorden civil" describes the whole process starting with a peaceful demonstration and ending as a violent confrontation or even civil war. The corresponding Russian and Bulgarian "массовые беспорядки"/"масови безредици" refer to the planned or spontaneous criminal actions of protesters, such as property damage, arsons, use of explosives, etc., as arising out of a peaceful action under certain circumstances (these terms are close to the English "mass disturbances, rioting" and Spanish "disturbios"). That is, all of these terms include the semantic component "illegal obstructive actions, rioting", nevertheless, their semantic coverage differs.

"Riot" ("mass_disorders", "mass_disturbances") is a "violent outburst of lawlessness by the people in a district" [25]. According to Wikipedia, it denotes a general violent disturbance against property, authority, people, etc.. The Russian and Bulgarian alignments are "бунт" (large-scale spontaneous armed uprising).

As it can be seen, the English term "riot" is closest in its meaning to the Bulgarian "масови безредици", Russian "массовые беспорядки" and Spanish "disturbios", whereas the Russian and Bulgarian "бунт" and Spanish "revuelta" are closest to "uprising" as they denote a large-scale violent response to authority's actions (e.g., "national uprising"), hovewer, they differ from the organized uprisings "въстание", "восстание" and "rebelión", because of their spontaneous nature.

A "mutiny" is an "open rebellion against a lawful authority (especially of soldiers and sailors)" [25]. English Wikipedia defines it as a criminal conspiracy by a group of people aiming to overthrow their superiors. The Bulgarian and French alignments define it as an open anarchic armed uprising against authorities in the military/marine domain. The Swedish "myteri" denotes police/military disobedience (primarily, non-violent) to the superiors and is considered as a treason by the state law.

A "revolt" is a general "rebellion or rising". A "rebellion" denotes the act of "rebelling, especially against a government"[25]. In Wikipedia, the English "rebellion", "revolt" and the French "révolte" are broad terms spanning rebellions, mutinies, civil disobedience, etc.. The Bulgarian and Swedish terms refer strictly to an open organized armed uprising, the Russian term spans uprising, riot, mutiny, and putsch.

As it can be seen, the terminology is not unified and has not been aligned properly across languages on Wikipedia. In order to clarify to some extent the difference between the terms for hierarchy building, we recur to such key event attributes as scale, spontaneity, and violence. As a result, we distinguish between street violent actions against people, property, or authority ("массовые беспорядки", "масови безредици", "disturbios", hereinafter "mass disorders, riots"), large-scale spontaneous uprisings ("бунт", "бунт", "revuelta", hereinafter "spontaneous uprising"), and large-scale organized uprisings ("въстание", "восстание", "rebelión", "organized uprising").

TABLE 4.1: Wikipedia alignment of the terms "civil disorder", "riot", "mutiny" and "rebellion"

Term|Language	**Bulgarian**	**French**	**Polish**	**Russian**	**Spanish**	**Swedish**
"civil disorder"	масовите безредици	N/A	N/A	массовые беспорядки	desorden civil	N/A
"riot"	бунт	émeute	zamieszki	бунт	disturbio	kravall
"mutiny"	метеж	mutinerie	N/A	N/A	motín	myteri
"rebellion" , "revolt"	въстание	révolte	powstanie	восстание	rebelión	uppror

These terms used in the political discourse are crucial for manipulating the opinion of the public, as observed in [13], and their use in modern press can reflect the intentions of the media to represent certain events in a positive or negative light.

Properties. Ontology classes are connected automatically via IS-A relations. Specific relations can be manually defined in Protégé by establishing an object property hierarchy, which currently includes a basic set of rules, such as HasAntecedent, HasConsequence, HasProperty (HasReason, IsInitiatedBy, IsLocatedIn, IsSupportedBy). Currently, event annotation uses only the class hierarchy.

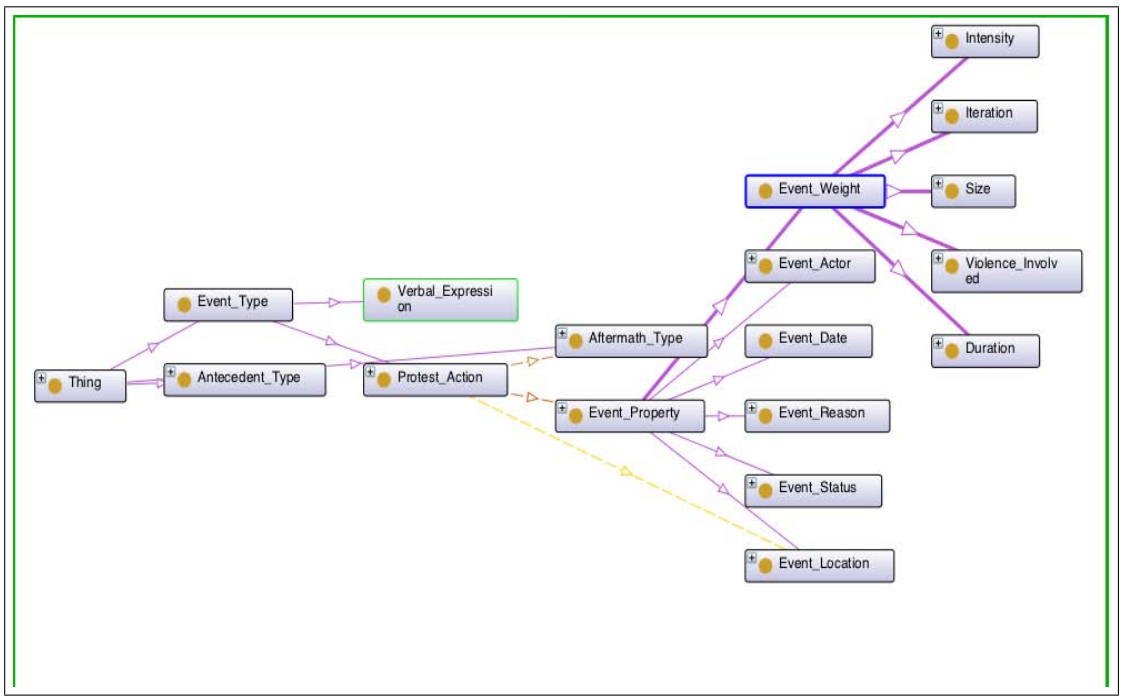

FIGURE 4.1: An excerpt from the social protest event hierarchy

4.2 Natural Language Representation of Information Slots: Gazetteers and Patterns

In the present work, we consider the analysis of very short text blocks (news headlines). The intuition behind the selection of this analysis unit is that headlines tend to contain compact and clear description of the key event concepts. As observed in [29], according to the journalistic "5W rule" that determines the press article structure, the answers to Who, What, Where, When and Why are commonly given in the lead sentences of a document. In [56], it is proved that in 80% of cases the noun in the title of a press article is the main argument of an event. Also, in [37] it is observed that processing titles has a number of advantages: there is no need to analyse coreference, because relevant facts are often summarized in a single sentence, and superficial patterns that are closer to the real data can be applied.

From the viewpoint of the functional grammar, a clause is represented by a configuration of semantic components, namely: process, participants and (optionally) circumstantial elements. Process is the central component, participants constitute the next layer, and circumstances - the third one. Circumstances add temporal, spatial, causal or other information to the central elements, but they are considered not directly involved in the process unlike participants. The typical representation of these components in

terms of the experiential structure is as follows: (i) process - verbal group, (ii) participant - nominal group, (iii) circumstance - adverbial group or prepositional phrase [19]. In our work we employ the term phrase instead of group, however, [19] sees the difference between a phrase and a group in that "a group is an expansion of a word" and "a phrase is a contraction of a clause". In Halliday's terminology, there are nominal, verbal, adverbial and conjunction groups, and prepositional phrases.

In news headlines related to contentious collective action, such as protest actions and violent attacks, the most frequent configurations are formed by the following semantic components: process (Event_Type or the type of collective action), participant (Event_Actor or people group), circumstantial elements (Event_Reason or the position of protesting groups towards an issue, and Event_Location or the physical geographical location, where a given event takes place). Event_Type, Event_Reason and Event_Location features are the main descriptors of a given event (protest action) and can be used for classification purposes. In [12], we have proved that these event scenario features enhance short-text (headlines) clustering accuracy. The natural language representation of these components together with five other circumstantial attributes that reflect the impact of a given issue of protest on the protesting groups (Event_Weight) is discussed in the present Section.

The use of semantic frames would make it relatively easy to fill protest event scenarios by annotating the roles around the event type expressed by a noun or a verb, however, currently, the FrameNet knowledge base cannot be used to draw out the data, because of two reasons: (1) there are few language-specific FrameNets, and these resources are not aligned to the main FrameNet, (2) the Frame Index does not contain many of the terms needed for detection of the ontology concepts (such as boycott, picket, riot, etc.).

4.2.1 Event_Type

Gazetteer. The task of Event_Type detection in this implementation consists in matching an almost exact substring (minimum edit distance = 0.15) or word lemma against the Event_Type gazetteer, connected to the ontology. The multilingual gazetteer currently contains the following parallel terms shown in Tab. 4.2. The correspondences were found using Wikipedia, Google Search, and our experimental corpus of news stories. The lexemes coming from the English "meeting" (the Bulgarian and Russian "митинг", the Spanish "mitin", the Polish "miting", and Swedish "möte") are used both as synonyms of demonstration, manifestation, rally and to denote a political (pre-electional) campaign. The word "crusade" that originally denotes military expeditions of Christians to liberate

the Holy Land (Jerusalem) from Muslim oppression, also, in a figurative sense, stands for a public campaign in defence of a cause[3].

The multilingual gazetteer of the Event_Type includes a lists of triggering terms aligned with the ontology concepts via eventmappings.def. Currently, the main triggers list covers almost all of the entries of the Event_Type ontology class with their non-English correspondences, except several specific job action types and symbolic acts. The gazetteer includes the following lists (semicolon separates list name, majorType and minorType features), as defined in the lists.def:

```
List name: majorType: minorType
boycott.lst:Type:boycott
escrache.lst:Type:escrache
picket.lst:Type:picket
sabotage.lst:Type:sabotage
blockade.lst:Type:blockade
sit-in.lst:Type:sit-in
demonstration.lst:Type:demonstration
march.lst:Type:march
protest.lst:Type:protest
hunger\_strike.lst:Type:hunger\_strike
strike.lst:Type:strike
casserole.lst:Type:casserole
flash\_mob.lst:Type:flash\_mob
riot.lst:Type:riot
spontaneous.lst:Type:spontaneous\_uprising
organized.lst:Type:organized\_uprising
insurgency.lst:Type:insurgency
mutiny.lst:Type:mutiny
coup.lst:Type:Coup\_d\_État
revolution.lst:Type:revolution
```

[3]Dictionnaire d'Académie Française online: http://atilf.atilf.fr/academie9.htm, the Oxford Dictionary of the English Language: http://oxforddictionaries.com/definition/english/, etc.

TABLE 4.2: Cross-lingual correspondences for the main ontology-based triggering terms

Keyword|Language	**Bulgarian**	**French**	**Polish**	**Russian**	**Spanish**	**Swedish**
protest	протест	protest, protestation, contestation	protest	протест	protesta	protest
boycott	бойкот	boycott	bojkot	бойкот	boicot	bojkott
demonstration	демонстрация	démonstration	demonstracja	демонстрация	demonstración	demonstration
manifestation	манифестация	manifestation, manif	manifestacja	манифестация, выступление	manifestación	manifestation
rally, meeting	митинг	rassemblement	rajd, miting	митинг	mitin, manifestación	rally, massmöte
casserole	протест с тигани (тенджери), протест с празни тигани (тенджери)	concert de casseroles	protesty z pustymi garnkami, marsz pustych garnków	кастрюльный марш, марш пустых кастрюль	cacerolada, cacerolazo, caceroleada, caceroleo	gryta, kastrull, kastrull-protest
flash mob	флашмоб	flash mob	flash mob	флешмоб	flashmob	flashmob
parade	парад	parade	parada	парад	desfile	parad
procession	шествие, процесия	procession	procesja, pochód	шествие, процессия	procesión	procession
march	марш	marche	marsz	марш	marcha	marsch
crusade (figurative sense)	кръстоносен поход	croisade	krucjata	крестовый поход	cruzada	korståg
picketing	пикетиране	piquet	pikieta	пикет	piquete	strejkvakt
sit-in	седящ протест/ демонстрация	sit-in	siedzący protest /demonstracja, etc.	сидячая забастовка/ демонстрация	sentada	sit-in, sittande protest/demonstration
strike	стачка	grève	strajk	забастовка	huelga	strejk
hunger strike	гладуване	grève de la faim	głodówka	голодовка	huelga de hambre	hungerstrejk
sabotage	саботаж	sabotage	sabotaż	саботаж	sabotaje	sabotage
riot	масови безредици	émeutes	zamieszki	массовые беспорядки	disturbios	kravall, uppiopp
(spontaneous) uprising	бунт	émeute	bunt	бунт	revuelta	revolt
(organized) rebellion, uprising, revolt	въстание	révolte, rébellion	powstanie, rebelia	восстание	rebelión, sublevación, levantamiento	uppror, revolt
insurgency	-	insurgence, insurrection	insurekcja	инсуррекция, инсургенция	insurgencia, insurrección	insurrektion
mutiny	метеж	mutinerie	-	мятеж	motín	myteri
Coup d'État, overthrow, coup	държавен преврат, преврат	Coup d'État, coup	zamach stanu, zamach	государственный переворот, переворот	golpe de estado	statskupp
putsch	пуч	putsch	pucz	путч	putsch	putsch
revolution	революция	revolution	rewolucja	революция	revolución	revolution

TABLE 4.3: Secondary triggering terms

Keyword|Language	**Bulgarian**	**French**	**Polish**	**Russian**	**Spanish**	**Swedish**
gathering	събрание	rassemblement, mobilisation	zgromadzenie	собрание, сход	concentración, movilización	samling
action	акция	action	akcja	акция	acción	aktion

The natural language expression of symbolic acts is too diverse, however, there is a simple pattern similar to <Symbolic_Act>Transitive_Verb + Object</Symbolic_Act> + "as a sign of protest" that can be used to extract the necessary substring. Out of job actions, we cover only strike and general_strike subtypes, because these terms characterize not only employer-workers, but also government-society interactions. The auxiliary triggers, such as "gathering" and "action" that do not themselves designate any particular event type, do often replace more specific terms (Tab. 4.3) and are included in

Table 4.4: Event type (noun phrase head)

Language	**Headline**
Bulgarian	[Протест]^T на майките в Харманли срещу бежанците. (Protest of mothers in Harmanli against refugees)
French	Hongrie: [manifestation]^T contre le régime du Premier ministre conservateur Viktor Orban. (Hungary: manifestation against the regime of Prime minister rightwing Viktor Orban)
Polish	Wielotysięczny [protest]^T w Tbilisi przeciw polityce Rosji. (Multi-thousand protest in Tbilisi against politics of Russia)
Russian	Братислава: [протест]^T против внешней политики Москвы. (Bratislava: protest against international policy of Moscow)
Spanish	[Concentración]^T en Alicante para apoyar a los estudiantes detenidos en Valencia. (Protest in Alicante to support the students arrested in Valencia)
Swedish	[Manifestation]^T i Tunis mot terrorn. (Manifestation in Tunis against terrorism)
English	Mass [protest]^T in Moscow against 'coup' in Kiev.

protest.lst: Oppositions hold mass gathering against government in the capital; Russian activists hold action against torture.

Patterns. The Event_Type description is parsed and annotated by ETgrammar in the JAPE grammar cascade. The type of a newly reported event commonly fills the positions of syntactic subject, predicate or object of the main clause.

The Event_Type is mainly represented by:

- a noun phrase, where the head matches one of the gazetteer entries (Tab. 4.4, where the uppercase T denotes "Trigger");
- a compound nominal predicate, where the nominal part matches one of the gazetteer entries (Tab. 4.5);
- a principal verb denoting a key concept (Tab. 4.6).

From now on, we provide metaphrases (literal translation) in round brackets for each non-English sentence in the tables, so that part-of-speech characteristics of each word can be visible. The English row contains a headline with similarly distributed concepts.

Our task here is to correctly detect the event type and assign the corresponding ontology class, creating a language-independent representation of event. At the first stage, the JAPE pattern/rule pair for the Event_Type slot is constructed as follows. In the LHS, the ontology-based lookup of the triggering terms is performed, and in the RHS, the corresponding "class" feature is assigned to the Event_Type annotation (see Fig. 4.2).

This simple rule together with BWP gazetteer runtime parameters (wholeWordsOnly = false, caseSensitive = false, normalizedDistanceThreshold = 0.15) allows to

TABLE 4.5: Event type (nominal part of a compound nominal predicate)

Language	**Headline**
Bulgarian	Европа на [протест]^T срещу TTIP. (Europe in protest against TTIP) САЩ: Хиляди медицински сестри на [стачка]^T заради ебола. (USA: Thousands of nurses on strike against Ebola)
French	Les Arabes israéliens en [grève]^T pour protester contre la destruction de maisons. (Israelian arabs on strike against the destruction of residences)
Polish	-
Russian	-
Spanish	Limpiadores de aviones en NY en [huelga]^T por temor a ébola. (Aircraft cleaners in NY on strike because of the fear of Ebola)
Swedish	Tusentals i [protest]^T mot polisfriande i New York. (Thousands in protest against the police release in New York) Bönder i Colombia i [strejk]^T mot jordbrukspolitik. (Colombian farmers on strike against the agricultural policy)
English	Radio France journalists on [strike]^T against budget cuts.

TABLE 4.6: Event type (principal verb)

Language	**Headline**
Bulgarian	Граждани ще [протестират]^T срещу здравен законопроект. (Citizens will protest against the draft law on healthcare services)
French	Des milliers de Russes [marchent]^T contre Poutine et pour la libération des prisonniers politiques. (Thousands of Russians march against Putin and for the liberation of political prisoners)
Polish	We Francji [strajkują]^T kolejarze i kierowcy autobusów. (In France protest railwaymen and bus drivers)
Russian	В Киеве в очередной раз [протестуют]^T против законопроекта "о мирных собраниях". (In Kiev once again (people) protest against the draft law on "peaceful gatherings")
Spanish	Feministas [piquetean]^T el Departamento de Protección a la Mujer. (Feminists are piqueting the Department for Women's Rights Protection)
Swedish	Sanaa: Tusentals [demonstrerar]^T mot Houthi-rebeller. (Sanaa: Thousands demonstrate against Houthi rebels)
English	Dozens injured as thousands [demonstrate]^T against racism and police brutality in Israel.

capture the event type in all of the above positions, as well as in compound nouns, noun phrases indicating unnamed groups (protesters, demonstrators), etc..

At the final stage, features of all of the annotations are mapped to the headline-based Protest_Event annotation. In case there are several annotations of event type contained in a sentence, the first one is selected to represent an event. This results beneficial, because, in most cases, the second mention is yet another description of the same event, or it can constitute a cause mention (Event_Reason). As it has been observed in [11], Event_Reason occupies the final position in the headline in a half of the dataset: $< Event\_Reason >< End\_Point >$ - 46%. In an earlier version of the pipeline [11] [12], we apply context-dependent annotation of event features, which is aimed at resolving ambiguity, but, when applied to many languages, it produces many false positives, and the recall drops drastically. Therefore, we opt for disambiguation at a later stage (Section 4.2.3).

FIGURE 4.2: Pattern/Rule pair for the ontology-based annotation of event type

```
Phase: ETdetector
Input: Token Lookup Split
Options: control = appelt

Rule: ET

(
 {Lookup.majorType == Type}
):et
-->
:et{
AnnotationSet etAS = (AnnotationSet) bindings.get("et");
AnnotationSet LookupAS = inputAS.get("Lookup",
    etAS.firstNode().getOffset(), etAS.lastNode().getOffset());

HashSet fNames = new HashSet();
fNames.add("class");
AnnotationSet ontoLookup = LookupAS.get("Lookup", fNames);
if (ontoLookup != null && ontoLookup.size()>0)
{
  Annotation  lookup   = ontoLookup.iterator().next();
  FeatureMap LookupFeatures = lookup.getFeatures();
  FeatureMap etFeatures = Factory.newFeatureMap();
  etFeatures.put("class", LookupFeatures.get("class"));
  etFeatures.put("ontology", LookupFeatures.get("ontology"));
  outputAS.add(LookupAS.firstNode(), LookupAS.lastNode(), "Event_Type",
    etFeatures);
}
}
```

4.2.2 Event_Reason

Event reason is one of the semantic components that most frequently accompany Event _Type and constitute one of the main descriptors of a protest event. Event reason is defined as a "Why" of an event. Headlines provide answers to this question by indicating the position of a protesting group over an issue. A protest is a public response of a civil group to the actions of governing authorities, a disagreement with the status quo. Protesters may show negative attitude towards a cause, defend a cause, express their demands over a cause, support someone or something in a conmemoration act. In some headlines, the issue is expressed via other relations (e.g., temporal) and the position of protesters is not defined explicitly.

In accordance with these observations, the present Section introduces gazetteer lists and considers several types of natural language representation patterns of event reason.

Gazetteer. The event reason annotation covers a fixed string corresponding to the Position of protesters (against, inSupportOf, forAndAgainst, toConmemorate, toDemand, or relatedTo), and a random string, which represents a specific claim, a cause, an

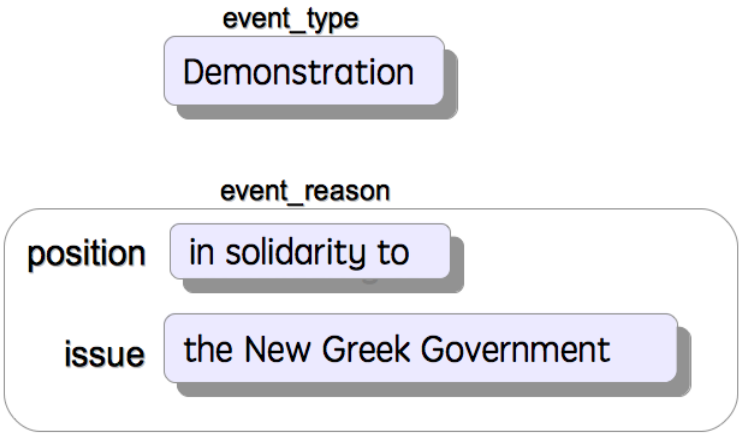

FIGURE 4.3: Event_Reason representation

entity, or other, and characterizes the Issue of protest (Financial, Ideological, Political, or other), as shown in Fig. 4.3.

The structure of the corresponding gazetteers connected to the ontology (via eventmappings.def) is as follows:

```
List name:majorType:minorType (Position)
actionProtest.lst:Reason:against
AsSign.lst:ReasonSymb:against
verbCause.lst:ReasonInv:against
actionSupport.lst:Reason:inSupportOf
nounCause.lst:ReasonAttr:inSupportOf
actionMemory.lst:Reason:toConmemorate
actionBoth.lst:Reason:forAndAgainst
actionClaim.lst:Reason:toDemand
actionClaimII.lst:ReasonDirObj:toDemand
actionMisc.lst:Reason:relatedTo
```

actionProtest.lst and AsSign.lst contain keywords that define the negative attitude of a civil group via relations: (i) AGAINST, (ii) IN RESPONSE TO, (iii) AS A REFUSAL TO. verbCause.lst lists triggering verbs that cover the actions, such as "to cause", "to trigger", "to produce", "to generate", "to spark", "to provoke", "to bring out to", etc.. The semantic subject of these verbs often denotes the issue of protest, while the inferred position is "against". Some of the lexemes corresponding to these concepts in Bulgarian, French, Polish, Russian, Spanish, and Swedish, are shown in Tab. 4.7.

TABLE 4.7: A partial list of verbs triggering the cause of an event in the position of semantic subject (based on the corpus and Google Search data)

Lexemes|Language	**Bulgarian**	**French**	**Polish**	**Russian**	**Spanish**	**Swedish**
cause	предизвиквам	susciter, entraîner	wywołać, powodować	вызывать, становиться причиной	causar	rendera, medföra
generate, produce, activate	отприщя	déclencher	budzić, potęgować		producir, generar	bevaka
spark, erupt	разискря			разжигать	disparar	
bring [Actor] out, push [Actor]	изкарвам, изведа	mener	popychać	толкать	empujar, sacar	
provoke		provoquer, inciter	prowokować	провоцировать	provocar	utlösa
lead to	доведа (до)	conduire (à)	doprowadzić (do)	приводить (к)	llevar (a)	leda (till)

The verbs like "to provoke" can have either inanimate or animate semantic subject in this context, which may lead to the misinterpretation of the semantic subject position content by the system: "Hunger strike by dervishes in Iran provokes protests" (event_reason) and "PKK leader Öcalan provoked Kobani protest, Deputy PM claims" (event_actor).

actionSupport.lst includes keywords for the positive attitude of protesters towards a cause: (i) IN FAVOR OF, (ii) IN SOLIDARITY TO, (iii) IN DEFENCE OF, (iv) IN SUPPORT OF. actionMemory.lst describes the relations (i) IN CONMEMORATION OF, (ii) IN HONOR OF, (iii) TO GLORIFY. These events are of interest, because they are closely related to the attitude of civil groups towards the government. In case a civil group's position is contradictory to the authority's one, a peaceful march may result in violent repressions. actionBoth.lst is used for cases, where protests arise both "for" and "against" a cause: "Hundreds in Tel Aviv protest for and against deportation plan".

actionClaim.lst and actionClaimII.lst trigger the mentions of specific demands expressed by protesting groups. Basically, together with conmemoration, it is another expression of support.

actionMisc.lst contains keywords for the relations that do not explicitly indicate the position of protesters: (i) AFTER, (ii) BECAUSE OF, (iii) ON THE OCCASION OF.

Patterns. The Event_Reason description is parsed and annotated by ERgrammar and further merged by ERmerge in the JAPE grammar cascade. Within the ERgrammar phase there are 6 macro patterns and 7 main pattern/rule pairs. Generic patterns for the Event_Reason circumstantial element are based on the conceptual structure of a sentence and morphosyntactic properties of its constituents. We distinguish between the following types of Event_Reason natural language representation:

- prepositional phrase (PrepPhrase rule);

TABLE 4.8: Event_Reason slot natural language representation: prepositional phrase

Language	**Headline**
Bulgarian	[Протест]^{T} на майките в Харманли [срещу бежанците]^{ER}. (Protest of mothers in Harmanli against refugees)
French	Hongrie : [manifestation]^{T} [contre le régime du Premier ministre conservateur Viktor Orban]^{ER}. (Hungary: manifestation against the regime of the conservative Prime Minister Viktor Orban)
Polish	Wielkie [demonstracje]^{T} w Genewie i Lozannie [przeciw szczytowi G8]^{ER}. (Big demonstrations in Geneve and Lozanne against the G8 summit)
Russian	В Баварии [протестуют]^{T} [против саммита G8]^{ER}. (In Bavaria (people) protest against the G8 summit)
Spanish	[Manifestación]^{T} en Moscú [contra Putin]ER y [de apoyo a Ucrania]^{ER}. (Manifestation in Moscow against Putin and in support of Ukraine)
Swedish	New Yorkbor [protesterade]^{T} [mot polisvåld]^{ER}. (New Yorkers protested against police violence)
English	Londoners are [protesting]^{T} [against the Ferguson decision]^{ER}.

- direct verb object (DirectObject and DirectObject-II rules);
- noun modifiers (NounModifier rule);
- semantic subject (SemanticSubject rule);
- subordinate clause of purpose and attributive clause (SubordinateClause rule).

(1) prepositional phrase: triggers: "Position" ontology lookup, Position= against, inSupportOf, etc..

The most frequent type of natural language representation observed in our experimental corpus of news headlines is prepositional phrase attached to Event_Type predicate or noun phrase, as in Tab. 4.8. Here, superscript ER denotes event reason and T - event trigger.

The schematic representation of the corresponding pattern (conceptual level and the underlying language structures) is depicted in Fig. 4.4. Here, the event type can be represented by either a noun phrase or a verb form and attaches a prepositional phrase that constitutes the event reason representation.

The PrepPhrase rule (Fig. 4.5) annotates prepositional phrases that include the issue and position of protesters. Here, CONTENT is a macro pattern that represents the protest issue text span. It can be any token (from 1 to 12), except for a finite verb (save Subjunctive mood) or a punctuation mark. Hereinafter, this macro together with the whole set of macros for the detection of information slots are specified in Appendix B. In step_1, the event type annotation is matched, which must precede the event reason and can be distanced by 0-20 tokens from event reason. In step_2, the "Position" value is looked up, and, in step_3, the issue content is matched. Step_4 serves to capture the

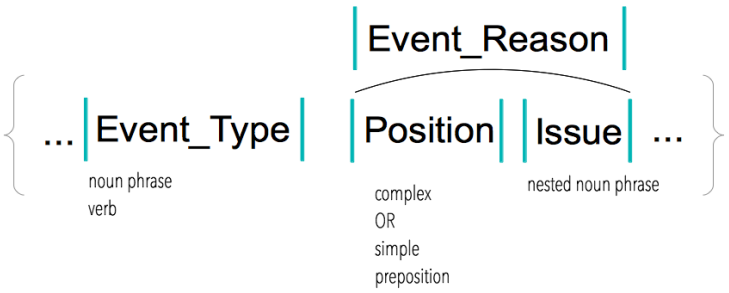

FIGURE 4.4: Event_Reason grammar: PrepPhrase pattern

mentions of both "for" and "against" positions in the same headline: e.g., "against Russia and for Ukraine". It allows to avoid the misannotation of "Russia and for Ukraine" as an issue of protest. In step_5, the ontology class features for "Position" are extracted from the Lookup annotations and assigned to the Event_Reason_L (left part) annotations. The corresponding strings are assigned to the Issue features of Event_Reason_R (right part) annotations.

(2) direct verb object: triggers: sabotage, boycott, demand, want etc., Position = against or toDemand.

The DirectObject pattern is depicted in Fig. 4.6. The round brackets denote the optional position of an element. Thus, the Event_Type component may occupy one of the three positions. For instance, in a phrase "Displaced Swat residents demand better living facilities in a protest in Islamabad" the structure will be as follows: Position ("demand") - Issue ("better living facilities") - Event_Type ("in a manifestation"), in "Ferguson protesters want their money" - Event_Type ("protesters") - Position ("want") - Issue ("their money").

The JAPE DirectObject rule is given in Fig. 4.7. Here, PREMOD is a macro pattern for noun premodifiers. It covers premodifiers expressed with an optional determiner (definite article) and an adjective/adjectives or participle/participles (from 1 to 2). The LHS lists three pattern options, according to the above structure. Each of the three patterns is composed of the same parts. In step_1, the pattern matches the Lookup annotation of verbs that describe the demands of protesting groups, which is the value of "Position" feature (here, "toDemand"). Next, the frequently appearing prepositional phrase "in a/the demonstration" or "during a/the demonstration" is annotated. The "Issue" feature is then highlighted. It can be optionally introduced with the conjunction

FIGURE 4.5: Jape pattern/rule pairs to identify event reason in the prepositional phrases attached to the verb object or subject

```
Rule: PrepPhrase

//step_1
(
{Event_Type}
({Token})[0,20]

//step_2
({Lookup.majorType == Reason}):pos

//step_3
(CONTENT):iss

//step_4
(
{Lookup.minorType == and}
({Lookup.majorType == Reason}):pos2
(CONTENT):iss2
)?
)

-->

//step_5
{
 AnnotationSet posAS = (AnnotationSet) bindings.get("pos");
 AnnotationSet LookupAS = inputAS.get("Lookup",
    issAS.firstNode().getOffset(), issAS.lastNode().getOffset());
 Annotation lookup = LookupAS.iterator().next();
 FeatureMap LookupFeatures = lookup.getFeatures();
 FeatureMap posFeatures = Factory.newFeatureMap();
 posFeatures.put("Position", LookupFeatures.get("class"));
 outputAS.add(LookupAS.firstNode(), LookupAS.lastNode(), "Event_Reason_L",
    posFeatures);
},
:iss.Event_Reason_R = {Issue = :iss@string},

:iss2{
 AnnotationSet posAS = (AnnotationSet) bindings.get("pos2");
 AnnotationSet LookupAS = inputAS.get("Lookup",
    issAS.firstNode().getOffset(), issAS.lastNode().getOffset());
 Annotation lookup = LookupAS.iterator().next();
 FeatureMap LookupFeatures = lookup.getFeatures();
 FeatureMap posFeatures = Factory.newFeatureMap();
 posFeatures.put("Position", LookupFeatures.get("class"));
 outputAS.add(LookupAS.firstNode(), LookupAS.lastNode(), "Event_Reason_L",
    posFeatures);
},
:iss2.Event_Reason_R = {Issue = :iss2@string}
```

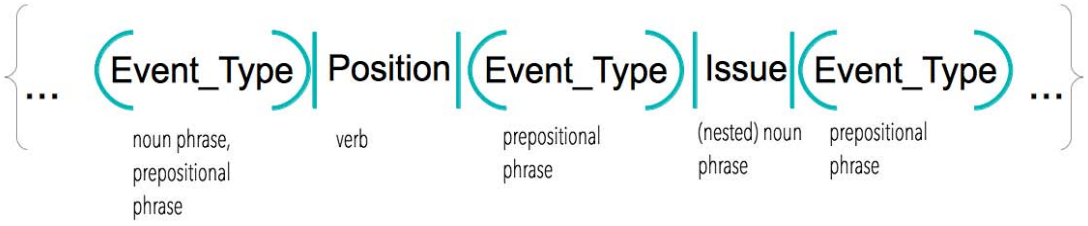

FIGURE 4.6: Event_Reason grammar: DirectObject pattern

TABLE 4.9: Event_Reason representation: direct verb object of the verbs "to boycott", "to sabotage"

Language	**Headline**
Bulgarian	[Бойкотират]^{T} [изборите]^{ER} в София заради отпадъците. ((People) are boycotting elections in Sofia because of the garbage)
French	L'opposition soudanaise persiste à [boycotter]^{T} [les élections]^{ER}. (Sudanese opposition persists in boycotting elections)
Polish	EU [bojkotuje]^{T} [proizvode iz Izraela]^{ER}. (EU is boycotting Israeli goods)
Russian	Британцы [бойкотируют]^{T} [французские товары]^{ER}. (British people are boycotting french goods)
Spanish	Una protesta estudiantil [boicotea]^{T} [la celebración del Claustro de la UB]^{ER}. (A student protest boycotts the UB Academic Board Meeting)
Swedish	Polacker [bojkottar]^{T} [Ikea]^{ER}. (Polish people boycott Ikea)
English	People are [boycotting]^{T} [United Airlines]^{ER} following an 'Islamophobic' incident.

"que" (in the Spanish language). In step_4, the corresponding labels are assigned to the strings via the action code.

The verbs boycott and sabotage denote opposition to some action. Sample sentences in the considered languages are given in Tab. 4.9. The "Position" component ("against") is already integrated into the verb semantics. For instance, the verb "to boycott" commonly attaches a noun phrase denoting the entity being boycotted: Norwegians are boycotting Israel. In this case, the "against" component is integrated into the verb semantics and does not need a separate representation via a preposition. The same is the case of the term "sabotage", where the direct object of the verb "to sabotage" stands for the activity being sabotaged: Unnamed animal rights activists are sabotaging products with cyanic acid. These are the cases, where the DirectObject-II applies. However, the corresponding nouns ("boycott", "sabotage") introduce event reason not only with the preposition ("of"), but also with "against", "in support of", etc. across the considered languages: e.g., Boycott against Israel. In this case, the above mentioned PrepPhrase rule is triggered. The schematic representation of the DirectObject-II pattern is given in Fig. 4.8.

In DirectObject-II pattern/rule pair (Fig. 4.9), step_1 matches the triggering verbs (boycott and sabotage) that attach event reason as a direct object. In step_2, the

FIGURE 4.7: Jape pattern/rule pairs to identify event reason in the position of direct verb object

```
Rule: DirectObject

//step_1
(
({Lookup.majorType == ReasonDirObj}):pos1
(
  ({Token.string ==~ "en|con"}|{Token.string ==~
    "durantena|"}|{Token.string ==~ "во время"})
  (PREMOD)?
  ({Event_Type})
)
({Token.string == "que"})?
(CONTENT):iss1
)
|

//step_2
(
(
  ({Token.string ==~ "en|con"}|{Token.string ==~
    "durantena|"}|{Token.string ==~ "во время"})
   (PREMOD)?
   ({Event_Type})
)
({Token})[0,7]
({Lookup.majorType == ReasonDirObj}):pos2
({Token.kind == word})[0,5]
(CONTENT):iss2
)
|

//step_3
(
({Lookup.majorType == ReasonDirObj}):pos3
({Token.kind == word})[0,5]
(CONTENT):iss3

({Token.kind == word})[0,5]
(
  ({Token.string ==~ "en|con"}|{Token.string ==~
    "durantena|"}|{Token.string ==~ "во время"})
  (PREMOD)?
  ({Event_Type})
)
)

-->

//step_4
:pos1.Event_Reason_L = {Position = toDemand},
:pos2.Event_Reason_L = {Position = toDemand},
:pos3.Event_Reason_L = {Position = toDemand},
:iss1.Event_Reason_R = {Issue = :iss1@string},
:iss2.Event_Reason_R = {Issue = :iss2@string},
:iss3.Event_Reason_R = {Issue = :iss3@string}
```

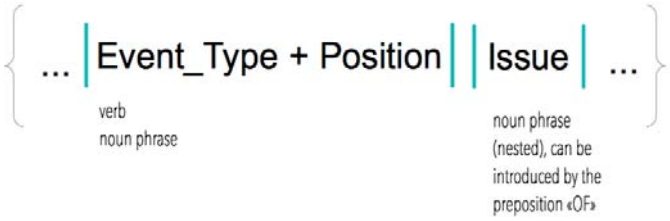

FIGURE 4.8: Event_Reason grammar: DirectObjectII pattern

FIGURE 4.9: Event_Reason grammar: DirectObject-II rule

```
Rule: DirectObject-II

//step_1
({Lookup.minorType == boycott}|{Lookup.minorType == sabotage})

//step_2
(CONTENT):er

-->

//step_3
:er.Event_Reason = {Position = against, Issue = :er@string}
```

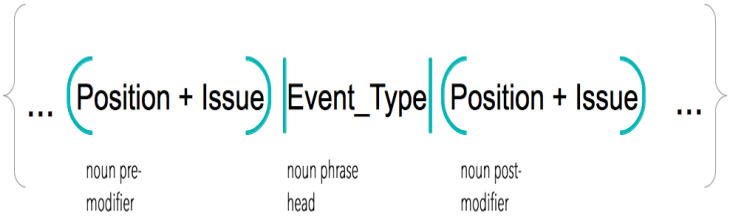

FIGURE 4.10: Event_Reason grammar: NounModifier pattern

respective issue (CONTENT macro) is covered. In step_3, the annotation is enriched with "Position" and "Issue" features.

(3) noun pre-modifier and post-modifier: triggers: anti- and pro- prefixes, Position = against or inSupportOf.

The NounModifier pattern serves to detect event reason in noun (event type) modifiers: "anti-government protests" (a modifier in preposition), "una protesta anti-nazi" (a modifier in postposition). Sample sentences for the considered languages are listed in Tab. 4.10. The schematic representation of the pattern is given in Fig. 4.10.

Table 4.10: Event_Reason representation: noun modifier

Language	**Headline**
Bulgarian	[Антиевропейски]^{ER} [пруски]^{ER} [протести]^{T} в Молдова. (Anti-European pro-Russian protests in Moldova)
French	[Manifestation]^{T} [anticapitaliste]^{ER} du 1er mai: environ 300 personnes ont été arrêtées. (Anti-capitalist manifestation of the 1st of May: around 300 people arrested)
Polish	[Antyrządowe]^{ER} [demonstracje]^{T} w Macedonii. (Anti-government demonstrations in Macedonia)
Russian	В Бахрейне возобновились [антиправительственные]^{ER} [митинги]^{T}. (In Bahrain renewed anti-government rallies)
Spanish	Nuevas [protestas]^{T} [anti Ahmadineyad]^{ER} en el treinta aniversario del asalto a la embajada americana. (New anti-Ahmadineyad protests in the 30th aniversary of the assault against the American embassy)
Swedish	Många deltog i [antivålds]^{ER}[manifestation]^{T}. (Many people participated in the anti-violence manifestation)
English	Hundreds of Ukrainian right-wingers hold [anti-government]^{ER} [rally]^{T} in Kiev.

The JAPE rules are shown in Fig. 4.11, where ANTI and PRO are macro patterns. ANTI serves for the adjectives starting with "anti(-)", "anty(-)" and "анти(-)" substrings, and PRO - with "pro" and "про" correspondingly. Stop lists are included to handle ambiguity issues. The NounModifier pattern matches event type annotations with possible pre- and postmodifiers that express positive or negative attitude of the civil groups towards a cause. In step_1, the macro patterns ANTI or PRO get matched (from 0 to 2 items). In step_2, the following token that must correspond to the event type annotation is covered. In step_3, the postmodifier containing one of the above mentioned substrings gets matched, if any. In step_4, the annotation is created and enriched with "Position" and "Issue" features.

(4) semantic subject: triggers: verbCause.lst lookup, Position = against.

The SemanticSubject pattern/rule pair serves to detect event reason mentions in the position of semantic subject: an issue causes a protest. The position of protesting groups towards a given issue in this case is inferred as negative. Sentence samples for the considered languages are given in Tab. 4.11. The schematic representation of the pattern is shown in Fig. 4.12.

In SemanticSubject pattern/rule pair (Fig. 4.13), the event reason precedes the triggering verb and event type annotation. The attitude towards the cause is negative ("Position = against"). Step_1 serves to match from 1 to 7 tokens before the triggering verb, which constitutes the string assigned to the Issue feature. In step_2, the triggering verb is looked up followed by 0 to 3 random tokens. Step_3 matches the event type noun phrase, which can be preceded by a token of up to 3 characters (a preposition). POSTMOD is a macro pattern for noun postmodifiers (adjectival/participial). Step_4 transduces the pattern into an annotation of type "Event_Reason" and assigns the corresponding features.

FIGURE 4.11: JAPE pattern/rule pairs to identify event reason in the noun modifiers

```
Rule: NounModifier

//step_1
(
(ANTI):antiI
|
(PRO):proI
)[0,2]

//step_2
{Event_Type}

//step_3
(
(ANTI):antiII
|
(PRO):proII
)?
-->

//step_4
:antiI.Event_Reason = {Position = against, Issue = :antiI@string},
:antiII.Event_Reason = {Position = against, Issue = :antiII@string},
:proI.Event_Reason = {Position = inSupportOf, Issue = :proI@string},
:proII.Event_Reason = {Position = inSupportOf, Issue = :proII@string}
```

TABLE 4.11: Event_Reason representation: semantic subject

Language	**Headline**
Bulgarian	[Забавени заплати]^{ER} изведоха енергетици на [протест]^{T}. (Delayed salaries brought power plant workers to protest)
French	[La fermeture des bases françaises]^{ER} suscite [protestations]^{T} au Sénégal. (The closure of French bases causes protests in Senegal)
Polish	[Wzrost cen paliw]^{ER} wywołał [protest]^{T} kierowców i rybaków w UE. (Fuel price increase caused a protest of drivers and fishermen in EU)
Russian	[В Германии очередное заседание саммита "Большой семерки"]^{ER} вызвало массу [протестов]^{T} среди мирного населения. (In Germany a new meeting of G7 members caused multiple civilian protests)
Spanish	[Ley laboral para jóvenes]^{ER} dispara [protestas]^{T} en Perú. (Youth labour law sparked protests in Peru)
Swedish	[Mladić gripande]^{ER} utlöste [protester]^{T}. (Mladić arrest provoked protests)
English	[The strikes]^{ER} sparked [protests]^{T} in Jordan.

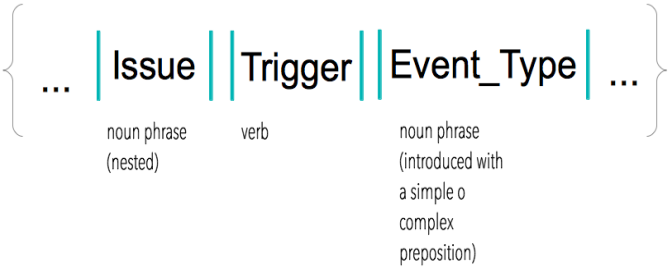

FIGURE 4.12: Event_Reason grammar: SemanticSubject pattern

FIGURE 4.13: JAPE pattern/rule pairs to identify event reason in the position of semantic subject

```
Rule: SemanticSubject

//step_1
(({Token})[1,7]):iss

//step_2
{Lookup.majorType == ReasonInv}
({Token})[0,3]

//step_3
({Token.length < 3})?
(PREMOD)?
{Lookup.majorType == Type}
(POSTMOD)?
-->

//step_4
:iss.Event_Reason = {Issue = :iss@string, Position = against}
```

(5) subordinate clause of purpose: triggers: Event_Type and purpattrConj.lst , Position = inSupportOf;

The SubordinateClause pattern/rule pair detects the event reason mentions in subordinate clauses of purpose. The rule covers two versions of the pattern: (i) to demand in a protest that something happens (the desired "happening" represents the event reason): "Residents of Corosal, Whiteland also demand that their roads be fixed in a protest at Whiteland Junction yesterday" and (ii) to protest to/in order to/with the aim of/etc. an event reason. Here, the position of protesting groups towards a given issue is inferred as positive (inSupportOf). The schematic pattern is depicted in Fig. 4.14 reflecting the two possible structures, where the star symbol denotes the optional positions of the event type prepositional phrase (e.g., "in a protest").

An excerpt of the JAPE rule is shown in Fig. 4.15. Step_1 block matches either an event type annotation with a postmodifier, if any, or a actionClaimII.lst lookup with the prepositional phrase "in a/the demonstration" or "during a/the demonstration", if any. In step_2, the list of subordinating conjunctions of purpose is looked up. It currently includes the following terms: за да, en vue de, pour que, afin que, afin de, żeby, dlatego że, с тем, чтобы, чтобы, para que, con el fin de que, för att. Step_3 matches the "Issue" feature, which constitutes any token (from 1 to 12), except for a punctuation mark. In Step_4, the :er pattern is transduced into an annotation.

Event_Reason merging phase. The ERmerge grammar is shown in Fig. 4.16. Here, in step_1, the left and right parts of the Event_Reason annotation are matched, and, in step_2, the "Position" feature is extracted from the left annotation and assigned

FIGURE 4.14: Event_Reason grammar: SubordinateClause pattern

FIGURE 4.15: JAPE pattern/rule pairs to identify event reason in the position of subordinate clause of purpose

```
Rule: SubordinateClause

//step_1
(
 ({Event_Type}
 (POSTMOD)?)
 |
 (
  {Lookup.majorType == ReasonDirObj}
  (
   ({Token.string ==~ "en|con"}|{Token.string ==~
    "durantена|"}|{Token.string ==~ "во время"})
   (PREMOD)?
   {Event_Type}
  )?
 )
)

//step_2
 {Lookup.minorType == purpattrConj}

//step_3
(
 {Token.kind == word}
 (
 {Token.kind != punctuation}
 )[0,12]
):er

-->

//step_4
:er.Event_Reason = {Position = inSupportOf, Issue = :er@string}
```

FIGURE 4.16: Event_Reason merging phase

```
Phase: ERmerge
Input: Event_Reason_L Event_Reason_R
Options: control = appelt

Rule: Merge

//step_1
(
{Event_Reason_L}
{Event_Reason_R}
):ann

-->

//step_2
:ann{

 gate.AnnotationSet annAS = (gate.AnnotationSet) bindings.get("ann");
 gate.Annotation ann = (gate.Annotation) annAS.iterator().next();
 FeatureMap annFeatures = ann.getFeatures();

 AnnotationSet erlAS = inputAS.get("Event_Reason_L").getContained(
    annAS.firstNode().getOffset(),
    annAS.lastNode().getOffset());
 for(Annotation erlAnn : erlAS)
  {
            FeatureMap erlFeatures  = erlAnn.getFeatures();
            annFeatures.put("Position", erlFeatures.get("Position"));
  }

//step_3
AnnotationSet errAS = inputAS.get("Event_Reason_R").getContained(
    annAS.firstNode().getOffset(),
    annAS.lastNode().getOffset());
 for(Annotation errAnn : errAS)
  {
            FeatureMap errFeatures  = errAnn.getFeatures();
            annFeatures.put("Issue", errFeatures.get("Issue"));
  }
try{
 outputAS.add(annAS.firstNode().getOffset(), annAS.lastNode().getOffset(),
    "Event_Reason", annFeatures);
}catch(InvalidOffsetException e){
   throw new LuckyException(e);
 }
 inputAS.remove(ann);
}
```

to the main Event_Reason annotation that covers both the left and right parts. In step_3, the "Issue" feature is extracted from the right part of the annotation and assigned to Event_Reason.

Disambiguation. In order to disambiguate between the mentions of event type, we rely on the Event_Reason grammars, particularly, the SemanticSubject rule. Here, the triggering predicate divides the sentence. In case, an event type mention or an ambiguous

term occurs on the left, it belongs to Event_Reason annotation, and the first one on the right is clearly Event_Type: "[Pride parades]^{ER} [caused]^{T} strong [protests]^{ET} in Russia" (where uppercase T stands for triggering verb, ET - event type, and ER - event reason).

4.2.3 Event Location

Gazetteers. From the perspective of natural language representation, in a news headline, event location semantics is most commonly carried by (1) noun phrases or prepositional phrases that may specify an administrative unit ("Moscow", "in Georgia", "in the city of London"), (2) noun and prepositional phrases that specify physical settings ("on the bridge of Buzaibz"), (3) nouns and adjectives defining the nationality a civil group ("indians", "Nigerian"). Gazetteers. In the current implementation, we use a combination of DBpedia semantic annotations and in-house gazetteers of geographical names to annotate Event_Location. euroPEA internal gazetteers are as follows:

```
LocataRU.lst:Location (3035 entries)
LocataES.lst:Location (409 entries)
LocataSE.lst:Location (88 entries)
LocataUNI.lst:Location (5284 entries)
LocataWorld.lst:Location:World (12 entries)
NatAdj.lst:Nationality (779 entries)
```

The Russian and "universal" gazetteer lists (LocataRU.lst, LocataUNI.lst) make use of the integrated GATE packages (ANNIE gazetteer and Lang_Russian) and in-house automatically collected gazetteer (for Russian). The "universal" list contains English and French location names that are partially adopted by most of the European countries. The Spanish and Swedish gazetteer lists are based on open online resources and focus on language-specific versions of location names: e.g., Storbritannien (Swedish, "Great Britain"), Spanien (Swedish, "Spain"), Ryssland (Swedish, "Russia"), Estados Unidos or EE. UU. (Spanish, "USA"), los Países Bajos (Spanish, "The Netherlands"), etc.. The mapping of gazetteer lists into our ontology classes has not been accomplished, because currently we rely on the access to the DBpedia remote repository via LKB gazetteer and use our in-house gazetteer as complementary. LocataWorld.lst is applied for the cases, where the whole world is defined as the location of protest. NatAdj.lst contains a list of nationality adjectives. The gazetteer of auxiliary phrases (AuxGazetteer) includes location prepositions and phrases (PrepLoc.lst) for the considered languages to be used in simple gazetteer-free patterns.

Our DBpedia query extracts labels for the classes: City, Settlement, Town, EthnicGroup, and properties: capital, demonym, language. It allows to deal with common location names, as well as with nationality nouns and adjectives. The united query with comments is shown in Fig. 4.17. An output of the query for the EthnicGroup class is partially depicted in Fig. 4.18.

Patterns. A set of patterns is created in order to deal with ambiguous location mentions based on DBpedia and our in-house gazetteer lookup. When the event type annotation precedes event reason in a sentence, the most reliable location mentions occur before the event reason information block: in a sentence like "Thousands of Russians have rallied in Moscow in protest against the armed conflict in Ukraine on Sunday" the token "Moscow" should be annotated as Event_Location. The nationality mention in this case refers to Russia, which is correct, however, such mentions can be confusing (e.g.: "Iranian communist women protest hijabs in Sweden"), therefore, we plan to perform disambiguation on the basis of metadata or the first paragraph of a given article. In cases, where the first location mention occurs after the Event_Reason_L annotation, ambiguity may arise, as in the following sentences:

- Civilians protest against the war *in Iraq in Great Britain.*
- Thousands protest against *Israel in Cape Town.*

A good-quality syntactic parser seems to be a solution, however, it would result in another dependency on foreign software, and it would have to be trained separately for each of the languages.

The pattern set relies on the gazetteer lookup, simple rules and discourse properties. Location mentions are being searched for within three text spans (Fig. 4.19): (i) before the Event_Type annotation (BeforeETann rule), (ii) between the Event_Type and Event_Reason annotations (BetweenETERann rule), (iii) after the Event_Reason annotation (AfterERann rule). The spans' lengths vary depending on position of Event_Type and Event_Reason annotations.

BeforeETann rule (Fig. 4.20) annotates event location mentions that are followed by Event_Type annotations in a given headline. In step_1, the first limitation is set: event location may be preceded by 0-20 tokens that cannot include Event_Reason. In step_2, DBpedia ontology, our in-house gazetteer lookup, or a simple :rule is matched, which includes a location preposition followed by 1-3 uppercase tokens (not a person name). Step_3 says that there can be a number of tokens (0-20) between the location and type annotations, however, this span must not contain the verbCause.lst lookup nor

FIGURE 4.17: A sample query to retrieve location names via the DBpedia endpoint

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX res:  <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?label ?x ?country
WHERE {

#step_1: a city name query
 { ?country rdf:type dbpedia-owl:Country .
   ?country dbpedia-owl:capital ?x .
   ?x rdfs:label ?label . }
UNION

#step_2: a city name query
  { ?x rdf:type dbpedia-owl:Settlement;
       rdfs:label ?label ;
       dbpedia-owl:country ?country. }
UNION


#step_3: a city name query
 { ?x rdf:type dbpedia-owl:Town;
      rdfs:label ?label ;
      dbpedia-owl:country ?country. }
UNION


#step_4: a city name query
{ ?x rdf:type dbpedia-owl:City;
     rdfs:label ?label ;
     dbpedia-owl:country ?country. }
UNION

#step_5: a query for the name of an ethnic group
{ ?x rdf:type dbpedia-owl:EthnicGroup;
     rdfs:label ?label . }
UNION

#step_6: a language name query
{ ?country a dbpedia-owl:Country .
  ?country dbpedia-owl:language ?x .
  ?x rdfs:label ?label .}
UNION

#step_7: a demonym query
{ ?country a dbpedia-owl:Country .
  ?country dbpedia-owl:demonym ?x .
  ?x  rdfs:label ?label.}
UNION

#step_8: a common country name query
{  ?x a dbpedia-owl:Country .
   ?x dbpedia-owl:demonym ?demonym ;
      rdfs:label ?label.}
}
```

"Russians"@en	http://dbpedia.org/resource/Russians
"روس"	http://dbpedia.org/resource/Russians
"Russen"@de	http://dbpedia.org/resource/Russians
"Pueblo ruso"@es	http://dbpedia.org/resource/Russians
"Russes"@fr	http://dbpedia.org/resource/Russians
"Russi"@it	http://dbpedia.org/resource/Russians
"ロシア人"@ja	http://dbpedia.org/resource/Russians
"Russen (volk)"	http://dbpedia.org/resource/Russians
"Rosjanie"	http://dbpedia.org/resource/Russians
"Russos"	http://dbpedia.org/resource/Russians
"Русские"@ru	http://dbpedia.org/resource/Russians
"俄羅斯人"@zh	http://dbpedia.org/resource/Russians
"Senufo people"@en	http://dbpedia.org/resource/Senufo_people
"Senufo (Volk)"@de	http://dbpedia.org/resource/Senufo_people
"Senufo"@es	http://dbpedia.org/resource/Senufo_people
"Sénoufos"@fr	http://dbpedia.org/resource/Senufo_people
"Senufo"@it	http://dbpedia.org/resource/Senufo_people
"Senufowie"	http://dbpedia.org/resource/Senufo_people
"Сенуфо (народ)"@ru	http://dbpedia.org/resource/Senufo_people
"Serbs"@en	http://dbpedia.org/resource/Serbs
"صرب"	http://dbpedia.org/resource/Serbs
"Serben"@de	http://dbpedia.org/resource/Serbs
"Serbios"@es	http://dbpedia.org/resource/Serbs
"Serbes"@fr	http://dbpedia.org/resource/Serbs
"Serbi"@it	http://dbpedia.org/resource/Serbs

FIGURE 4.18: A screenshot of DBpedia output for the EthnicGroup class

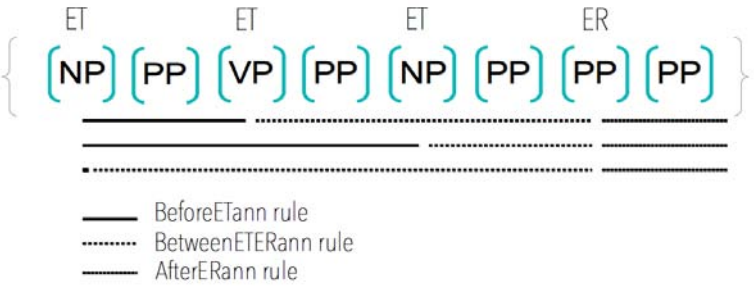

FIGURE 4.19: A schematic representation of the Event_Location pattern set

the prepositions, such as "regardless" or "after" (ExclPrep.lst). Step_4 matches the event type annotation.

In accordance with this rule, Event_Location annotations will be made as follows (superscript ET denotes Event Type):

- (1: Bulgarian) В <Event_Location Location_String = "Харманли">Харманли </Event_Location> [протестуват]^{ET} срещу бежанците (Mothers protest in Harmanli against refugees).

FIGURE 4.20: An experimental pattern to detect and extract Event_Location that occurs before Event_Type in a given headline

```
Rule: BeforeETann

//step_1
({Token, !Event_Reason})[0,20]

//step_2
({Lookup.minorType == locprep})?
(
//DBpedia: Location name and Ethnic Group annotation. Part of speech: Noun
    or Adjective
({Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "N|subst"}|
{Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "ADJ|adj"}
):dbpedia
|
//A simple pattern that annotates the following sequence: location
    preposition, from one to three tokens with uppercase initial
    characters.
//The tokens must not occupy the same span as person name annotation to
    avoid errors.
(
{Lookup.minorType == locprep}
(({Token.orth == "upperInitial", !Lookup.majorType !=
    person_first|person_full})[1,3]):rule
)
|
//The in-house GeoGazetteer lookup of location names, where the tokens
    must have the initial characters in uppercase
({Lookup.majorType == Location}
|
//The in-house GeoGazetteer lookup of nationality names
{Lookup.majorType == Nationality}):geogazetteer
)

//step_3
({Token, !Lookup.majorType == ReasonInv, !Lookup.minorType ==
    ExclPrep})[0,20]

//step_4
{Event_Type}
```

- (2: French) <Event_Location Location_Class: http://www.w3.org/2002/07/owl#Thing, Location_Instance: http://dbpedia.org/resource/Hungary, Location_String = "Hongrie">Hongrie</Event_Location> : [manifestation]^{ET} contre le régime du Premier ministre conservateur Viktor Orban (Hungary: a manifestation against the regime of the conservative prime minister Viktor Orban).

The Harmanli location name is not present in DBpedia, therefore, it is triggered by the :rule pattern.

FIGURE 4.21: Event_Location annotation between Event_Type and Event_Reason

```
Rule: BetweenETERann


//step_1
{Event_Type}
(POSTMOD)?
({Token})[0,5]

//step_2
{Lookup.minorType == locprep}
(PREMOD)?
(
({Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "N|subst"}|
{Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "ADJ|adj"}):dbpedia
|
(({Token.orth == upperInitial, !Lookup.majorType !=
    person_first|person_full})[1,3]):rule
|
({Lookup.majorType == Location, Token.orth == upperInitial}|
{Lookup.majorType == location}|
{Lookup.majorType == Nationality}):geogazetteer
)

//step_3
({Token})[0,5]
{Event_Reason}
```

BetweenETERann rule (Fig. 4.21) annotates event location in between the type and reason labels. In step_1, it is set that the Event_Type and the Event_Location prepositional phrase can be separated by a postmodifier and a maximum of 5 token annotations. In step_2, the Event_Location prepositional phrase is matched that is constructed of the same elements as in the above rule. Step_3 says the maximum distance from Event_Location to Event_Reason can be of 5 tokens.

In accordance with this rule, Event_Location annotations will be made as follows (superscript ET denotes Event Type, ER - Event Reason):

- (1: Bulgarian) Нов [протест]^{ET} в <Event_Location Location_String = "Будапеща" >Будапеща</Event_Location>, този път [срещу корупцията]^{ER} (A new protest in Budapest, this time against corruption).
- (2: Spanish) [Concentración]^{ET} en <Event_Location Location_Class: `http://dbpedia.org/resource/Spain`,Location_Instance:`http://dbpedia.org/resource/Alicante`, Location_String:"Alicante">Alicante</Event_Location> [para apoyar a los estudiantes detenidos en Valencia]^{ER} (A mobilization in Alicante in support of the students arrested in Valencia).

- (3: Spanish) [Protestas]T en <Event_Location Location_Class: `http://www.w3.org/2002/07/owl#Thing`, Location_Instance = `http://dbpedia.org/resource/Egypt`, Location_String = "Egipto">Egipto</Event_Location> [tras el nombramiento de un presunto exterrorista al frente de Luxor]^{ER} (Protests in Egypt after the appointment of a presumed former terrorist as governor of Luxor).

In DBpedia, there is a lack of Bulgarian labels for location names, therefore, in (3), location mention is triggered by the :rule pattern.

AfterERann rule (Fig. 4.22) creates Event_Location annotations after the Event_Reason lookup. In step_1, the Event_Reason annotations or lookup entries get matched, which allows to capture location mentions overlapping with the event reason content (e.g.: Thousands protest [against the war in Israel in Cape Town]^{ER}; Russians protest [against the armed conflict in Ukraine in Moscow]^{ER}). In step_2, the first Event_Location prepositional phrase pattern is optionally matched, in case there are two such phrases in a given headline. These phrases may be separated by a number of tokens (up to 5). In step_3, the main prepositional phrase is matched, which in most cases ends the sentence.

In accordance with this rule, Event_Location annotations will be made as follows (superscript ERT denotes Event Reason Trigger):

- (1: Bulgarian) Протести [заради]^{ERT} бомбения атентат в <Event_Location Location_Class: `http://www.w3.org/2002/07/owl#Thing`, Location_Instance = `http://dbpedia.org/resource/India`, Location_String = "Индия">Индия </Event_Location> (Protests against a bombing attack in India).
- (2: Spanish) Concentración [de solidaridad con]^{ERT} París en <Event_Location Location_Class: `http://dbpedia.org/resource/United_Arab_Republic`, Location_Instance = `http://dbpedia.org/resource/Cairo`, Location_String = "Cairo" >El Cairo </Event_Location> (A mobilization in solidarity to Paris in Cairo).
- (3: Swedish) Nakna kvinnor protesterade [mot]^{ERT} Egypten i <Event_Location Location_Class: `http://dbpedia.org/resource/Sweden`, Location_Instance = `http://dbpedia.org/resource/Stockholm`, Location_String = "Stockholm"> Stockholm </Event_Location> (Naked women protested against Egypt in Stockholm).

The action code converts each pattern into an annotation with the features "Location_Class" (country name), "Location_Instance" (city, nation, country name), "Location_String" (exact textual representation).

FIGURE 4.22: Event_Location annotation in the Event_Reason - <Sentence_End> interval

```
Rule: AfterERann

//step_1
({Event_Reason}|{Lookup.majorType == Reason})
({Token}) [0,10]

//step_2
(
{Lookup.minorType == locprep}
(PREMOD)?
(
({Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "N|subst"}|
{Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "ADJ|adj"})
|
(({Token.orth == upperInitial, !Lookup.majorType != person_first,
    !Lookup.majorType != person_full}) [1,3])
|
({Lookup.majorType == Location, Token.orth ==
    upperInitial}|{Lookup.majorType == location})
)
)?

//step_3
({Token.string ==~ "ввъв|"}|
{Token.string ==~ "во|en"}|
{Token.string ==~ "au|aux"}|
{Token.string ==~ "à|w"}|
{Token.string ==~ "pod|i"}|
{Token.string ==~ "åp"})
(PREMOD)?
(
({Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "N|subst"}|
{Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "ADJ|adj"}):dbpedia
|
(({Token.orth == upperInitial, !Lookup.majorType != person_first,
    !Lookup.majorType != person_full}) [1,3]):rule
|
({Lookup.majorType == Location, Token.orth ==
    upperInitial}|{Lookup.majorType == location}):geogazetteer
)
```

FIGURE 4.23: Event_Location as a prepositional phrase attached to the Event_Type

```
Rule: IndepAnnot

//step_1
{Event_Type}

//step_2
({Token, Lookup.majorType != Reason})[0,7]

//step_3
{Lookup.minorType == locprep}
(PREMOD)?
(
({Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "N|subst"}|
{Lookup.inst =~ "http://dbpedia.org/resource/", Token.category =~
    "ADJ|adj"}):dbpedia
|
(({Token.orth == upperInitial, !Lookup.majorType !=
    person_first|person_full})[1,3]):rule
|
({Lookup.majorType == Location, Token.orth == upperInitial}|
{Lookup.majorType == location}|
{Lookup.majorType == Nationality}):geogazetteer
)
```

The IndepAnnot rule (Fig. 4.23) contains an alternative pattern/rule pair for the creation of an Event_Location label that matches the corresponding prepositional phrase attached to the noun phrase (Event_Type) anywhere in the sentence. In step_1, event type is matched. Step_2 covers tokens that may separate the Event_Type noun phrase from the Event_Location prepositional phrase, where the possibility of Event_Reason lookup annotations is excluded to avoid ambiguity. In step_3, the core location patterns are matched, namely: dbpedia lookup, simple :rule, or gazetteer lookup.

In accordance with this rule, Event_Location annotations will be made as follows (superscript ET denotes Event Type):

- (1: Bulgarian) 26-има ранени при [протести]^{ET} в <Event_Location Location_Class: `http://www.w3.org/2002/07/owl#Thing`, Location_Instance = `http://dbpedia.org/resource/Spain`, Location_String = "Испания">Испания</Event_Location> (26 injured in protests in Spain).
- (2: French) La fermeture des bases françaises suscite des [protestations]^{ET} au <Event_Location Location_Class: `http://www.w3.org/2002/07/owl#Thing`, Location_Instance = `http://dbpedia.org/resource/Senegal`, Location_String = "Sénégal">Sénégal</Event_Location> (The closure of France's bases causes protests in Senegal).

- (3: Spanish) Las lágrimas de un activista impulsan las [protestas]^{ET} en <Event _Location Location_Class: `http://www.w3.org/2002/07/owl#Thing`, Location _Instance = `http://dbpedia.org/resource/Egypt`, Location_String = "Egipto"> Egipto</Event_Location> (Tears of an activist provoke protests in Egypt).

4.2.4 Protest Weight

Within the framework of this thesis, the term Protest Weight reflects the level of support of a given claim (Event_Reason) by protesting groups. It allows to detect long-term, iterative, increasing, massive, and violent events. The corresponding features are as follows: event Duration, Iteration, Intensity, Size and Violence_Involved. It equals 1 if at least one of the features is mentioned in a given headline, and 'null', if not. These features are extracted on the basis of in-house gazetteers using JAPE-Plus Extended patterns[4].

Gazetteer. The gazetteer of Protest_Weight contains several sublists per each of the features, according to the parts of speech and semantics of the corresponding key terms.

Duration indicates the time an action lasts, and can be specified in an exact number of days, weeks, months or years. The cases, where there are adverbial, adjectival, or verbal attributes indicating a lasting action, but no specification of an exact duration (e.g.: "people are still protesting"), the feature that should be annotated is the Event_Status (In_Progress). The part of the EventWeightGazetteer that deals with the event duration is presented with sample entries as follows.

```
List name:majorType:minorType
durationVerb.lst:Duration:durationVerb
продължавам:::bg
se poursuivre:::fr
trwać:::pl
продолжаться:::ru
continuar:::es
fortsätta:::se
...
stopVerb.lst:Duration:stopVerb
стихвам:::bg
cesser:::fr
```

[4]https://code.google.com/p/gateplugin-japeutils/

```
przestać:::pl
стихать:::ru
cesar:::es
slutar:::se
...
```

The durationVerb.lst contains synonyms of the verbs "to last", "to continue", and the stopVerb.lst list - synonyms of the verbs "to stop", "to cease", "to grow feeble".

Several lists from the auxiliary terms gazetteer (AuxGazetteer) are also used to detect the duration of an event:

```
List name:majorType:minorType
negationList.lst:Auxiliary:negation (a list of negative particles)
numNoun.lst:Numeral:numNoun (a list of numerical nouns,
such as "tens", "hundreds", "thousands", etc.)
date_unit.lst:Date_unit (a list of date unit nouns,
such as "day", "week", "month", etc.)
```

Iteration feature shows that a claim has been defended in consecutive actions. The part of the EventWeightGazetteer that deals with the event iteration is presented with the corresponding sample entries as follows.

```
List name:majorType:minorType
reAdv.lst:Iteration:reAdv
пак:::bg
de nouveau:::fr
znowu:::pl
снова:::ru
otra vez:::es
igen:::se
...
reTrigger.lst:Iteration:reTrigger
пореден:::bg
nouveau:::fr
kolejny:::pl
очередной:::ru
nuevo:::es
ny:::se
```

```
...
reVerb.lst:Iteration:reVerb
възобновя:::bg
reprendre:::fr
wznowić:::pl
возобновить:::ru
reanudar:::es
återvända:::se
...
```

reAdv.lst contains a list of adverbial attributes (e.g., "again", "one more time") that accompany the event type, when the latter is represented by a simple predicate, compound predicate, or a noun phrase head. reTrigger.lst includes adjectival attributes that form part of the event type noun phrase and are synonymous to the English "new" and "another". reVerb.lst lists verbs that add event type noun phrase as a direct object and are synonymous to "renew", "recommence".

Event Size may be specified in an exact number of participants, which requires the use of auxiliary number gazetteers, and unspecified (a "massive protest"). The part of the EventWeightGazetteer that deals with the event size is presented with the corresponding sample entries as follows.

```
List name:majorType:minorType
sizeAdv.lst:Size:sizeAdv
също:::bg
aussi:::fr
również:::pl
также:::ru
también:::es
med:::se
...
sizeAttr.lst:Size:sizeAttr
масов:::bg
massif:::fr
masowy:::pl
массовый:::ru
masivo:::es
mass:::se
...
```

```
sizeSeries.lst:Size:sizeSeries
вълна:::bg
vague:::fr
fala:::pl
волна:::ru
ola:::es
våg:::se
...
sizeSupport.lst:Size:sizeSupport
включа:::bg
se joigner:::fr
do␣łączić:::pl
подключиться:::ru
adherirse:::es
förenas:::se
...
```

sizeAdv.lst contains a list of adverbs similar to "also" that accompany the event type predicate. sizeAttr.lst includes attributive adjectives, coordinated or uncoordinated, appended to event type, that are synonymous to "multiple", "massive", "large". sizeSeries.lst lists nouns that commonly occupy the position of the head in an event type noun phrase and are synonymous to "wave", "series". sizeSupport.lst is a list triggering the subevent of support, where a civil group joins a protest, and, consequently, the size of the event increases. The entries are thus synonymous to "support" and "join".

Violence_Involved feature shows whether a protest or protests are or have become violent, in case the corresponding mentions are found in a given headline. The part of the EventWeightGazetteer that deals with the violence presence is shown with the corresponding sample entries as follows.

```
List name:majorType:minorType
violenceAdj.lst:Violence:violenceAdj
насилствен:::bg
violent:::fr
brutalny:::pl
ожесточенный:::ru
violento:::es
våldsam:::se
...
```

```
violenceNoun.lst:Violence:violenceNoun
насилие:::bg
violence:::fr
gwa␣lt:::pl
насилие:::ru
violencia:::es
våld:::se
...
```

violenceAdj.lst contains a list of adjectives similar to "violent", "radical", "aggressive" that enter the event type noun phrase or form part of the predicate. violenceNoun.lst includes nouns synonymous to "violence", "brawl" that denote violent attacks. This list also covers weapon names at the current stage. The gazetteer of auxiliary terms (AuxGazetteer) is used to capture the change of state (protests that become violent or turn into a riot):

```
List name:majorType:minorType
changeState.lst:Auxiliary:become
ставам:::bg
devenir:::fr
stać się:::pl
становиться:::ru
tornarse:::es
bli:::se
...
```

Intensity feature indicates that a given event is heating up, becomes larger and more intensive, as mentioned by the news. The part of the EventWeightGazetteer that deals with the event intensity is presented with the corresponding sample entries as follows.

```
List name:majorType:minorType
intensityAdj.lst:Intensity:intensityAdj
силен:::bg
fort:::fr
mocny:::pl
сильный:::ru
fuerte:::es
```

```
stark:::se
...
intensityCirc.lst:Intensity:intensityCirc
из целия свят:::bg
dans le monde entier:::fr
w ca␣lym świecie:::pl
во всем мире:::ru
en todo el mundo:::es
på hela världen:::se
...
intensityVerb.lst:Intensity:intensityVerb
засилвам:::bg
s'intensifier:::fr
nasilać się:::pl
усиливаться:::ru
intensificarse:::es
intensifieras:::se
...
```

intensityAdj.lst is a list of adjectival attributes similar to "strong", "hot", "powerful", and "intense" that accompany the event type noun phrase or form part of the corresponding compound nominal predicate. intensityCirc.lst includes circumstantial attributes that are appended to event type, and are synonymous to the expression "in the whole country/world/...". intensityVerb.lst lists verbs that connect to the event type noun phrase as a subject and are synonymous to "intensify", "strengthen", "grow".

Patterns. The present subsection shows sample annotations of protest weight together with the corresponding pattern/action rule pairs. Duration: EWgrammarI.jape. The first part of the event weight grammar describes the annotation of the exact protest duration. In the current version, we obtain strings as feature values, however, we plan to unify multilingual strings using our ontology. The sample labeled sentences are shown in Tab. 4.12. The number of days, weeks, etc. can be represented by a number ("12 dzień"/"12 day") a word, cardinal or ordinal numbers ("cinquième jour"/"fifth day"), or a combination of numbers, puntuation and alphabetic characters ("57-и ден"/"57th day").

The grammar (Fig. 4.24) includes three main steps: macro pattern definition, DurationSpecifiedI rule description and DurationSpecifiedII rule description.

FIGURE 4.24: Generic JAPE rules to detect the duration D of an event T (trigger), where the duration is specified in the number n of days/weeks/months

```
//step_1
Macro: DURATION

(
 (
  ({Lookup.majorType == Numeral}|{Token.kind == number})
  ({Token.length < 4})[0,2]
  {Lookup.majorType == Date_unit}
 )
 |
 (
  {Lookup.majorType == Date_unit}
  ({Lookup.majorType == Numeral}|{Token.kind == number})
  ({Token.length < 4})[0,2]
 )
)

//step_2
Rule: DurationSpecifiedI

(
(DURATION):ed
({Token.length < 3})?
(EVENT_TYPE)
)
|
(
(EVENT_TYPE)
({Token.length < 3})?
(DURATION):ed2
)
-->
:ed.Event_Duration = {Duration = :ed@string, EW = 1},
:ed2.Event_Duration = {Duration = :ed2@string, EW = 1}

//step_3
Rule: DurationSpecifiedII

((DURATION):ed1|(EVENT_TYPE))
({Token})[0,7]
(
 {Lookup.minorType == durationVerb}|
 ({Lookup.minorType == negation}{Lookup.minorType == stopVerb})
)
({Token})[0,7]
((DURATION):ed2|(EVENT_TYPE))

-->
:ed1.Event_Duration = {Duration = :ed1@string, EW = 1},
:ed2.Event_Duration = {Duration = :ed2@string, EW = 1}
```

TABLE 4.12: Event duration indicators position with respect to event type. The superscript D denotes event duration, T - event type

Language	**Headline**
Bulgarian	[57-и ден]^D на антиправителствени [протести]^T. (57th day of anti-government protests)
French	[Cinquième jour]^D de [manifestations]^T anti-américaines. (Fifth day of anti-american manifestations)
Polish	[12 dzień]^D [protestu]^T w JSW: strajk górników nie wygasa. (12 day of protest in JSW: miners' strike does not end)
Russian	[Демонстрации]^T в Нью-Йорке продолжаются [пятый день]^D. (Demonstrations in New York continue fifth day)
Spanish	[Segundo día]^D de [huelga]^T de trenes en París. (Second day of strike of trains in Paris)
Swedish	Demonstranter i New York inledde på söndagen en [femte dag]^D av [protester]^T mot överdrivet våld av poliser mot minoriteter. (Demonstrators in New York commence on Sunday the fifth day of protests against the excessive violence use by police against minorities)
English	It's the [11th day]^D of [unrest]^T [Protests]^T last for [20 days]^D. [Protest]^T do not stop for [20 days]^D. For [10 days]^D, [protests]^T continued in Selma.

In step_1, the DURATION macro chooses between two sequences of the same components: (i) an amount indicator (numeral or number), (ii) from 0 to 2 Tokens less than 4 characters long (a preposition) and (iii) the name of the date unit (day, week, month, year). In step_2, a substring similar to "2nd week of protests" gets matched and labeled. In step_3, sequences like "protests last/do not stop during 8 days" or "8 days last/do not stop protests" are captured.

Iteration: EWgrammarII.jape. The second part of the event weight grammar describes the situation, where protesters' gatherings in support of a certain claim are repeated consecutively. Sample labeled sentences are presented in Tab. 4.13. Here, the attribute "new" in a noun (event type) modifier position is an indicator of an action iteration. The event iteration indicator can be also expressed with an adverb (e.g., "again"), as in Tab. 4.14, or a verb (e.g., "renew"), as in Tab. 4.15, which is reflected in the corresponding gazetteer lists.

The EWgrammarII.jape grammar contains the following rules covering these cases. In Fig. 4.25 the rule detecting event iteration on the basis of reTrigger.lst list of Event_Type noun modifiers is depicted. It chooses between the sequences consisting of the following components: an event iteration trigger (reTrigger.lst) (noun modifier) and EVENT_TYPE macro.

In case the trigger is expressed via an adverb or an adverbial expression, Event_IterationII rule applies, which is based on the reAdv.lst list lookup (Fig. 4.26). We rely on the presence of the trigger regardless of its position in the sentence.

The verb triggers are captured by Event_IterationIII rule (Fig. 4.27). A reVerb.lst precedes the EVENT_TYPE macro pattern. The distance between these elements is set

TABLE 4.13: Event iteration indicators position with respect to the Event_Type on the basis of reTrigger.lst. The superscript RE denotes event iteration, T - event type

Language	**Headline**
Bulgarian	[Нов]^{RE} антиимигрантски [митинг]^{T} „срещу религиозния фанатизъм" в Дрезден. (New anti-immigrant rally "against religious fanatism" in Dresden)
French	[Nouvelle]^{RE} [grève]^{T} nationale en Grèce contre les mesures d'austérité. (New national strike in Greece against austerity measures)
Polish	[Nowa]^{RE} [demonstracja]^{T} przeciwko Saakaszwilemu w Tbilisi. (New demonstration against Saakashvili in Tbilisi)
Russian	Московские медики вышли на [новый]^{RE} [протест]^{T} против планов власти. (Moscow medical workers went out on a new protest against the plans of the government)
Spanish	La oposición proeuropea se enfrenta a la policía en [nuevas]^{RE} [protestas]^{T} contra el Gobierno en Ucrania. (Pro-European opposition clashes with the police in new protests against the Government in Ukraine)
Swedish	[Nya]^{RE} [protester]^{T} mot minskade kulturtidskriftstödet. (New protests against the reduction of cultural journalism funding)
English	Greek workers in [new]^{RE} [protests]^{T} against cuts.

TABLE 4.14: Event iteration indicators position with respect to Event_Type on the basis of reAdv.lst. The superscript RE denotes event iteration, T - event type

Language	**Headline**
Bulgarian	[Пак]^{RE} [протест]^{T} заради паркирането в София. (Again protest because of parking in Sofia)
hline French	Les routiers grecs [de nouveau]^{RE} en [grève]^{T} illimitée. (Greek truck drivers again on unlimited strike)
Polish	Rolnicy [ponownie]^{RE} [protestują]^{T} pod Biłgorajem. (Farmers again protest near Bilgoray)
Russian	В Киеве [в очередной раз]^{RE} [протестуют]^{T} против законопроекта "о мирных собраниях". (In Kiev once again protest against the draft law "on peaceful gatherings")
Spanish	Inspectores laborales [otra vez]^{RE} van a la [huelga]^{T}. (Labour inspectors again go on strike)
Swedish	Greker [åter]^{RE} ut i general[strejk]^{T}. (Greeks again out on a general strike)
English	Students [demonstrate]^{T} against increased tuition fees [again]^{RE}.

TABLE 4.15: Event iteration indicators position with respect to Event_Type on the basis of reVerb.lst. The superscript RE denotes event iteration, T - event type

Language	**Headline**
Bulgarian	От "Ремотекс" - [възобновяват]^{RE} [протестите]^{T} си. ((people) from "Remotex" renew their protests)
French	Loi de Santé : des médecins veulent [reprendre]^{RE} la [grève]^{T} des gardes. (Public Health Law: health workers want to recommence the "strike of guards")
Polish	Tybetańczycy [wznowili]^{RE} [protesty]^{T} w Katmandu. (Tibetans renewed protests in Katmandu)
Russian	В Грузии [возобновляют]^{RE} [протесты]^{T} против президента. (In Georgia (people) renew protests against the president)
Spanish	Españoles [vuelven a [protestar]^{T}]^{RE} contra privatización de la sanidad. (Spanish people are back to protest against the privatization of health services)
Swedish	Regimkritiker [återvänder för [att protestera]^{T}]^{RE}. (Regime critics return to their protests)
English	Latin Americans [renew]^{RE} their [protest]^{T} against Israel.

FIGURE 4.25: A rule to detect the iteration RE of an event T on the basis of reTrigger.lst

```
Rule: Event_Iteration
(
 (
  ({Lookup.minorType == reTrigger})
  (EVENT_TYPE)
 )
 |
 (
  (EVENT_TYPE)
  ({Lookup.minorType == reTrigger})
 )
):ei

-->
:ei.Event_Iteration = {Iteration = "True", EW = 1}
```

FIGURE 4.26: A rule to detect the iteration RE of an event T on the basis of reAdv.lst

```
Rule: Event_IterationII

({Lookup.minorType == reAdv}):ei

-->
:ei.Event_Iteration = {Iteration = "True", EW = 1}
```

FIGURE 4.27: A rule to detect the iteration RE of an event T on the basis of reVerb.lst

```
Rule: Event_IterationIII

(
  {Lookup.minorType == reVerb}
  ({Token.kind == word})[0,4]
  (EVENT_TYPE)
):ei

-->
:ei.Event_Iteration = {Iteration = "True", EW = 1}
```

to up to 4 tokens.

Size: EWgrammarIII.jape. The third part of the event weight grammar detects the size of an event/events. The number of participants may be specified (e.g., "7 thousand") or unspecified (e.g., "massive", "multithousand", "mega"). A protest may have been supported by a protesting group, a social movement organization or other, which increases the event weight. For these cases, the rules SizeSpecified, SizeNotSpecified (3) and Support_Event apply. A gazetteer and a rule have been also elaborated to detect the series of events (Event_Series). We do not focus on multiple events indicated by the plural form of Event_Type noun, because the solution is rather straightforward. The detection depends solely on the quality of part-of-speech taggers integrated into

TABLE 4.16: Event size indicators position with respect to event type (SizeSpecified rule). The superscript S denotes event size, T - event type

Language	**Headline**
Bulgarian	[7 хиляди]^S на [протест]^T в Гърция срещу новите мерки за икономии. (7 thousand on protest in Greece against new austerity measures)
French	Pologne: [un millier]^S de mineurs en [grève]^T contre la restructuration de leur compagnie. (Poland: a thousand miners on strike against the restructuring of their company)
Polish	[40 tysięcy]^S studentów [protestuje]^T przeciw podwyżce czesnego. (40 thousand students protest against the rise in tuition fees)
Russian	Более [110 тыс]^S человек пришли на [митинг-концерт]^T в поддержку Крыма на Красную площадь. (More than 110 thousand people came to the rally-concert in support of Crimea in the Red Square)
Spanish	Unos [2,5 millones]^S de turcos han [protestado]^T contra el gobierno, según Interior. (Around 2.5 millions of turks have protested against the government, according to Interior)
Swedish	[100 tusen]^S i [protest]^T mot Belgiens regering. (100 thousand in protest against the Belgium government)
English	[Eleven thousand]^S people [protest]^T against ETA violence in Vitoria.

the pipeline. The currently employed French and Spanish taggers from the Treetagger package provide bare part-of-speech tags with no additional morphological information, therefore, another tagger's output (e.g., Freeling) should be used.

The SizeSpecified pattern/rule pair serves to match and extract the mentions of the exact number of participants as shown in Tab. 4.16. The amount can be represented by a number, a word or a combination of both, which is the most frequent.

The pattern with the corresponding action code is depicted in Fig. 4.28. Step_1 matches the substring denoting the exact number participants. It covers the following representation forms: e.g., 55.000 OR 55,000 OR 55000 OR 55 000 OR 55 thousand OR fifty five thousand OR thousands. In step_2, the exact number mention is followed by 3 random Tokens and the Event_Type that can be either a verb or nominal part of a compound nominal predicate. Step_3 executes the action code.

Obviously, we do not attach the EW (event weight) feature to these annotations, because the identified size does not necessarily indicate a massive event. A post-filtering outside GATE can be done to sort out massive events with the exact number of participants.

The SizeNotSpecified-I, SizeNotSpecified-II and SizeNotSpecified-III pattern/rule pairs aim at detecting massive events with multiple participants without exact specifications. The trigger can be expressed either via a numerical noun ("thousands"), a noun modifier ("a multithousand protest"), or a compound noun, as in "megaprotest" (Tables 4.17 and 4.18):

The corresponding patterns with the action code are presented in Fig. 4.29. In step_1 one of the following three sequences is selected: (ii) an event size attribute

FIGURE 4.28: SizeSpecified rule to detect the size S of an event T

```
Rule: SizeSpecified

\\step_1
(
(
(
  (
  (
  {Token.kind == number}
  {Token.string ==~ "[.,]"}
  {Token.kind == number}
  )
  |
  ({Token.kind == number})[2]
  )
  ({Lookup.minorType == numNoun})?
)
|
(
  ({Token.kind != punctuation})?
  ({Token.kind == number}|{Lookup.minorType == number})
  {Lookup.minorType == numNoun}
)
):es1
|
(
 ({Token.kind == number}):es2
 {Token.kind != punctuation, Token.kind != number, !Lookup.minorType ==
    numNoun}
)
)

\\step_2
({Token})[0,3]

(
 ({Lookup.majorType == Type, Token.category =~ "VL"}|
 {Lookup.majorType == Type, Token.category =~ "Vp[ip][it]f"}|
 {Lookup.majorType == Type, Token.category =~ "praet"}|
 {Lookup.majorType == Type, Token.category =~ "fin"}|
 {Lookup.majorType == Type, Token.category =~ "Vmi[spf]"})
 |
 (
 ({Token.category =~ "VL"}|
 {Token.category =~ "Vp[ip][it]f"}|
 {Token.category =~ "praet"}|
 {Token.category =~ "fin"}|
 {Token.category =~ "Vmi[spf]"})?
({Token})[0,3]
({Token.string ==~ "en|i"}|{Token.string ==~ "нав|"}|{Token.string ==~
    "na|w"})
(PREMOD)?
({Lookup.majorType == Type, Token.category =~ "N|S"})
)
)

-->

\\step_3
:es1.Event_Size = {Size = :es1@string},
:es2.Event_Size = {Size = :es2@string}
```

TABLE 4.17: Event size indicators position with respect to event type (SizeNotSpecified-I rule). The superscript S denotes event size, T - event type

Language	**Headline**
Bulgarian	[Хиляди]^S [протестираха]^T в Брюксел срещу войната в Ирак. (Thousands protested in Brussels against the war in Iraq)
French	Des [milliers]^S de Russes [marchent]^T contre Poutine et pour la libération des prisonniers politiques. (Thousands of Russians march against Putin and for the liberation of political prisoners)
Polish	[Tysiące]^S Hiszpanów [protestują]^T przeciwko "paktowi na rzecz euro". (Thousands of Spaniards protest against the "pact for the euro")
Russian	[Десятки тысяч]^S британцев [требуют на демонстрациях]^T повышения зарплат. (Tens of thousands of Brits demand in demonstrations salary increase)
Spanish	[Miles]^S de personas [marchan]^T en Getafe contra las reformas de la Sanidad. (Thousands of people march in Getafe against the reforms in Health sector)
Swedish	[Tusentals]^S [protesterar]^T mot valfusk i Teheran. (Thousands protest against the electoral fraud in Teheran)
English	[Thousands]^S in [protest]^T against ECB Governing Council meeting in Cyprus.

TABLE 4.18: Event size indicators position with respect to event type (SizeNotSpecified-I and SizeNotSpecified-II rules). The superscript S denotes event size, T - event type

Language	**Headline**
Bulgarian	[Многохиляден]^S [митинг]^T в Одеса в памет на загиналите в Киев полицаи от Беркут. (Multithousand rally in Odessa in memory of Berkut officers killed in Kiev)
French	[Manifestation]^T [massive]^S à Rome contre la réforme du marché du travail. (Massive manifestation in Rome against the reform of the labour market)
Polish	Pakistan: [wielotysięczny]^S [protest]^T przeciwko "Charlie Hebdo" w Karaczi. (Pakistan: multithousand protest against "Charlie Hebdo" in Karaczi)
Russian	В центре Севастополя прошел [многотысячный]^S [митинг]^T участников «Русской весны». (In the centre of Sevastopol there was held a multithousand rally of the participants of "Russian spring")
Spanish	[Mega]^S[protesta]^T en Casablanca contra el plan de austeridad. (Megaprotest in Casablanca against the austerity plan)
Swedish	[Mass]^S[protest]^T mot Berlusconi i Rom. (Massprotest against Berlusconi in Rome)
English	London erupts in [mass]^S [protest]^T against Israeli crimes.

(sizeAttr.lst lookup), followed by the EVENT_TYPE macro (e.g., "multithousand protest"); (iii) EVENT_TYPE macro followed by the event size attribute (sizeAttr.lst lookup) (e.g., "protesta multitudinaria"). In step_2, the primary detection of compound nouns containing substrings "mega", "mass", etc. is performed, and a temporal annotation Event_Size_Temp is created. In step_3, a rule similar to SizeSpecified is described that detects mentions of numerical nouns, such as "thousands", "tens of thousands", "millions". The first restriction is that a numerical noun (numNoun.lst lookup) must not be preceded by a number. In what follows the rule is identical to SizeSpecified.

The Event_Series pattern/rule pair aims at capturing consecutive events occuring in different locations and supporting the same claim. The triggering expressions are similar to the English "a wave of", "a series of", "a storm of", "a chain of", "a flurry of", "a groundswell of", "a howl of", "a stir of", "a roar of", "a landslide of", "a stream of", "a torrent of", "a firestorm of", "a flood of", "a barrage of", etc.. Sample labeled sentences

FIGURE 4.29: SizeNotSpecified-I, SizeNotSpecified-II and SizeNotSpecified-III rules to detect the size S of an event T

```
\\step_1
Rule: SizeNotSpecified-I

(
    ({Lookup.minorType == sizeAttr}):es1
    (EVENT_TYPE)
)
|
(
   (EVENT_TYPE)
    ({Lookup.minorType == sizeAttr}):es2
)

-->
:es1.Event_Size = {Size = "multiple_participants", EW = 1},
:es2.Event_Size = {Size = "multiple_participants", EW = 1}

\\step_2
Rule: SizeNotSpecified-II

(
 {Token.string =~ "[Mm]ega|[Ss]tor"}|
 {Token.string =~ "[Mm]ass|[Jjä]tte"}
):es

-->
:es.Event_Size_Temp = {}

\\step_3
Rule: SizeNotSpecified-III

(
({Token.kind != number, !Lookup.minorType == number})?
({Lookup.minorType == numNoun}):es
)

({Token})[0,3]

(
 ({Lookup.majorType == Type, Token.category =~ "VL"}|
 {Lookup.majorType == Type, Token.category =~ "Vp[ip][it]f"}|
 {Lookup.majorType == Type, Token.category =~ "praet"}|
 {Lookup.majorType == Type, Token.category =~ "fin"}|
 {Lookup.majorType == Type, Token.category =~ "Vmi[spf]"})
 |
 (
 ({Token.category =~ "VL"}|
 {Token.category =~ "Vp[ip][it]f"}|
 {Token.category =~ "praet"}|
 {Token.category =~ "fin"}|
 {Token.category =~ "Vmi[spf]"})?
({Token})[0,3]
({Token.string ==~ "en|i"}|{Token.string ==~ "нав|"}|{Token.string ==~
    "na|w"})
(PREMOD)?
({Lookup.majorType == Type, Token.category =~ "N|S"})
)
)

-->
:es.Event_Size = {String = :es@string, Size = "multiple_participants", EW
    = 1}
```

TABLE 4.19: Event size indicators position with respect to event type (Event_Series rule). The superscript S denotes event size, T - event type

Language	**Headline**
Bulgarian	[Протестна]^T [вълна]^S срещу спирането на влакове в цялата страна. (A wave of protests against the train stoppage in the whole country)
French	Une [vague]^S de [protestations]^T contre le Mondial dans des grandes villes du Brésil. (A wave of protests against the World Cup in the major cities of Brazil)
Polish	Przez Niemcy przetacza się [fala]^S [protestów]^T przeciwko „islamizacji Europy". (Through Germany rolls a wave of protests against the "islamization of Europe")
Russian	В Бразилии прошла [волна]^S [забастовок]^T против ЧМ по футболу. (In Brazil there was a wave of strikes against the futbol World Cup)
Spanish	Una [ola]^S de [huelgas]^T trastorna Europa. (A wave of strikes sweeps Europe)
Swedish	En [våg]^S av [protester]^T mot Sverigedemokraterna sköljer över Sverige. (A wave of protests against Swedish democrats spreads over Sweden)
English	[Protest]^T [wave]^S against new Austrian government rises.

FIGURE 4.30: Size_Series rule to detect the size S of an event T

```
Rule: Event_Series

(
  {Event_Type, Token.kind == word}
  ({Lookup.minorType == sizeSeries}):es1
)
|
(
({Lookup.minorType == sizeSeries}):es2
({Token.length < 4})?
(EVENT_TYPE)
)

-->
:es1.Event_Size = {Size = "consecutive_events", EW = 1},
:es2.Event_Size = {Size = "consecutive_events", EW = 1}
```

are given in Tab. 4.19. The rule is depicted in Fig. 4.30. The pattern matches one of the following sequences: (i) Event_Type followed by the sizeSeries.lst (trigger) lookup, as in "protest wave", and (ii) sizeSeries.lst (trigger) lookup followed by an EVENT_TYPE macro with a preposition (Token of up to 4 characters length), as in "a wave of protests".

The Support_Event rule detects mentions of protest support by unnamed people groups, social movements, etc.. The support event trigger can be represented either by a verb (e.g., "join") (see Tab. 4.20) or an adverb (e.g., "also") (see Tab. 4.21). The rule (Fig. 4.31) covers three optional sequences: (i) step_1: trigger (sizeSupport.lst entry) precedes the Event_Type noun; (ii) step_2: trigger (sizeSupport.lst entry) follows the Event_Type noun; (iii) step_3: trigger (sizeAdv.lst entry) precedes the EVENT_TYPE macro.

Violence_Involved: EWgrammarIV.jape. The Violence_Involved rule aims at detecting mentions of violence use by non-governmental actors or authorities in the course

TABLE 4.20: Event size indicators position with respect to event type (Support_Event rule). The superscript S denotes event size, T - event type

Language	**Headline**
Bulgarian	Варненци се [включват]^S в [протест]^T срещу ескалиращото насилие над животни. (Varna citizens join the protest against the increasing violence towards animals)
French	Les Indigenes se [joignent]^S aux [manifestations]^T en Brésil. (Indigenes join the manifestations in Brazil)
Polish	Coraz więcej górników [dołącza]^S do [protestu]^T przeciw restrukturyzacji Kompanii Węglowej. (More and more miners join the protest against the restructuring of the Coal Company)
Russian	Члены "Коми войтыр" [поддерживают]^S [протест]^T против отмены обязательного изучения нацязыков. ("Komi voityr" members support the protest against the cancellation of the mandatory study of national languages)
Spanish	Miles de policías británicos se [unen]^S a la [protesta]^T contra los recortes. (Thousands of British police officers join the protest against cuts)
Swedish	Sjuksköterskor i Turkiet [stödjer]^S [protest]^T mot avsked för fackligt arbete. (Nurses in Turkey support the protest against the dismissal for union work)
English	40 000 [join]^S Cape Town [protest]^T against Israeli attacks.

TABLE 4.21: Event size indicators position with respect to event type (Support_Event rule). The superscript S denotes event size, T - event type

Language	**Headline**
Bulgarian	Майките [също]^S на [протест]^T - срещу новата такса за градини и ясли. (Mothers also on protest - against a new tax for kindergartens and public nurseries)
French	Les retraités [aussi]^S [protestent]^T pour leur pouvoir d'achat. (Pensioners also protest for their purhasing power)
Polish	Czesi [też]^S [protestują]^T przeciwko umowie ACTA. (Czechs also protest against the ACTA agreement)
Russian	Сотрудники Армянских электросетей [также]^S [протестуют]^T против обязательной накопительной компоненты пенсионной системы. (Armenian electrical network workers also protest against the defined contribution plan)
Spanish	Los jardineros de parques históricos de Madrid [también]^S en [huelga]^T. (Gardeners of the historical parks of Madrid also on strike)
Swedish	Ryssland: Anarkister [med]^S i [protest]^T mot Putin. (Russia: Anarchists also in protest against Putin)
English	In Cairo people [also]^S [protest]^T against new president Mursi.

of protest actions. The natural language triggers covered by our gazetteers currently include noun modifiers (e.g., "violent", "bloody", "fierce") and nouns (e.g., "violence", "outrage") (Tab. 4.22). The rule presented in Fig.4.32 chooses between two sequences: (i) from 0 to 5 tokens that do not start at the same offset with Type.lst and Reason.lst lookups, triggering Violence.lst lookup, from 0 to 5 random tokens and Event_Type annotation coinciding with a noun part-of-speech tag; (ii) Event_Type noun, from 0 to 5 tokens that do not coincide with the Reason.lst lookup, triggering Violence.lst lookup.

Intensity. EWgrammarIV.jape. The Event_Intensity rule aims at capturing protest events of increasing intensity. Triggers are mainly represented by verbs with the corresponding semantics (Tab. 4.23). The pattern (Fig. 4.33) matches one of the sequences depending on whether the Intensity.lst trigger precedes or follows the EVENT_TYPE macro pattern.

FIGURE 4.31: Support_Event rule to detect the size S of an event T expressed in the subevent of support

```
Rule: Support_Event

\\step_1
(
({Lookup.minorType == sizeSupport}):es1
({Token})[0,2]
({Token.length < 4})?
({Event_Type, Token.category =~ "N|subst"})
)
|

\\step_2
(
({Event_Type, Token.category =~ "N|subst"})
({Token})[0,2]
({Lookup.minorType == sizeSupport}):es2
)
|

\\step_3
(
({Lookup.minorType == sizeAdv}):es3
({Token.length < 4})?
(EVENT_TYPE)
)

-->
:es1.Event_Size = {Size = "increasing", Subevent = "support", EW = 1},
:es2.Event_Size = {Size = "increasing", Subevent = "support", EW = 1},
:es3.Event_Size = {Size = "increasing", Subevent = "support", EW = 1}
```

TABLE 4.22: Violence use indicators position with respect to Event_Type (Violence_Involved rule). The superscript V denotes the mention of violence use, T - event type

Language	**Headline**
Bulgarian	[Насилствени]^{V} [протести]^{T} в Афганистан срещу антиислямския филм. (Violent protests in Afghanistan against an anti-Islam film)
French	[Manifestations]^{T} [sanglantes]^{V} dans la ville irakienne de Kout. (Bloody protests in the Iraqi town of Kout)
Polish	W Turcji [gwałtowne]^{V} [protesty]^{T} przeciw forsowaniu szkół religijnych. (In Turkey violent protests against forcing religious schools)
Russian	По всей Франции прошли [ожесточённые]^{V} [протесты]^{T} против неоправданного применения силы со стороны полиции. (In the whole France took place violent protests against the unwarranted police violence)
Spanish	[Violencia]^{V} en Argelia en [protesta]^{T} por la crisis económica. (Violence in Algeria in protest related to the economic recession)
Swedish	[Våldsamma]^{V} [protester]^{T} i Hongkong. (Violent protests in Hong Kong)
English	[Violent]^{V} [protests]^{T} in Burma. [Violence]^{V} in [protests]^{T} against Congo election delay.

FIGURE 4.32: A generic JAPE rule to detect the presence of violence V in an event T

```
Rule: Violence_Involved

(
({Token, Lookup.majorType != Type, Lookup.majorType != Reason})[0,5]
({Lookup.majorType == Violence}):ev
({Token})[0,5]
{Event_Type, Token.category =~ "N|subst"}
)
|
(
{Event_Type, Token.category =~ "N|subst"}
({Token, Lookup.majorType != Reason})[0,5]
({Lookup.majorType == Violence}):ev2
)

-->
:ev.Event_Violence = {Violence_Use = "True", EW = 1},
:ev2.Event_Violence = {Violence_Use = "True", EW = 1}
```

TABLE 4.23: Event intensity indicators position with respect to event type (Event_Intensity rule). The superscript I denotes the mention of event intensity, T - event type

Language	**Headline**
Bulgarian	[Протестите]^T срещу ислямизацията в Германия [се засилват]^I. (Protests against the islamization in Germany strengthen)
French	Burkina Faso : des [manifestations]^T [s'intensifient]^I contre l'initiative du referendum constitutionnel. (Burkina Faso: manifestations intensify against the initiative of the constitutional referendum)
Polish	W USA [nasilają się]^I [protesty]^T przeciw meczetom. (In US strengthen protests against mosques)
Russian	[Протесты]^T против Путина в России [усиливаются]^I. (Protests against Putin in Russia grow)
Spanish	Las [manifestaciones]^T contra Conga [se intensifican]^I en la capital y en el departamento de Cajamarca. (Manifestations against Conga intensify in the capital and the department of Cajamarca)
Swedish	I Venezuela [intensifieras]^I [protesterna]^T mot president Nicolas Maduro. (In Venezuela intensify protests against the president Nicolas Maduro)
English	Student [protests]^T against austerity cuts [intensify]^I in Quebec.

FIGURE 4.33: A generic JAPE rule to detect events with an increasing intensity level

```
Rule: Event_Intensity

(
  (
   {Lookup.majorType == Intensity}
   ({Token})[0,2]
   (EVENT_TYPE)
  )
  |
  (
   (EVENT_TYPE)
   ({Token})[0,2]
   {Lookup.majorType == Intensity}
  )
):ei

-->
:ei.Event_Intensity = {Intensity = "Increasing", EW = 1}
```

Chapter 5

Event Collection and Extraction Evaluation

The evaluation of the automatic protest event selection and information blocks annotation has been performed by a multilingual domain expert. In future we plan to make use of multiple annotators' responses. Current resources do not allow us to have several multilingual experts or unexperienced annotators for each of the languages.

5.1 Event Data Collection Quality Evaluation

For protest event selection task, we present the counts of the total number of messages, extracted by crawlers per session, and the number of messages after the filtering of total headline duplicates and stopwords. The crawlers are run with the settings shown in Appendix D, where crawling domains and lists of triggering and co-occurring concepts are specified for each of the languages (Bulgarian, French, Polish, Russian, Spanish, Swedish). In the current implementation, trigger (Event_Type) lists cover the introduced event type ontology only in part. A document is considered a relevant search result, if it mentions an actual protest action identifiable from its attributes, such as event type, location, reason, date, actor and scale. As it can be seen in Tab. 5.1, the main portion is filtered out as total duplicates. Tab. 5.2 shows the number of manually checked instances per language after total duplicates filtering, the number of true negative reports, as well as the percentage of true positives.

The number of true positives is lower for the French language, because the corresponding query contains more ambiguous substrings (”croisade”, ”marche”, ”blockage”,

TABLE 5.1: The number of documents (headlines) in the language-specific datasets before and after the total duplicates filtering and stopwords removal

Language	**Before filtering**	**After total duplicates filering**	**After stopwords removal**
Bulgarian	4113 (528 kb)	1308	1306
French	8286 (615 kb)	1468	1242
Polish	5820 (591 kb)	1644	1561
Russian	7686 (673 kb)	4656	4654
Spanish	8678 (756 kb)	4683	4252
Swedish	3580 (180 kb)	705	695

TABLE 5.2: The percentage of the reports that have been manually sorted out as unrelated to the topic, and the substrings that represent the most ambiguous terms

Language	**checked, no.**	**filtered out, no.**	**filtered out, %**	**ambiguous terms**
Bulgarian	700	8	1	"шестви", "марш"
French	700	159	22	"croisade", "marche", "blockage", "concentration", "contestation", "rassemblement"
Polish	700	114	16	"blokow", "blokuj" "demonstracj"
Russian	700	55	8	"марш", "акци", "шестви", "выступ"
Spanish	700	93	13	"marcha", "concentraci",
Swedish	700	22	3	"blockeras", "manifestera", "marsch"

"concentration", "contestation", "rassemblement"), than queries elaborated for other languages. The majority of irrelevant reports belongs to the following domains: sport, music and traffic accidents.

5.2 Evaluation Datasets

For the purposes of annotation performance evaluation, 2 datasets have been formed on the basis of the crawled headlines. The development set contains 575 information-rich multilingual headlines (90-100 per language) that are used both for runtime settings tuning, patterns enhancement and performance evaluation. The test set includes 3000 multilingual headlines (500 per language) randomly selected from the main dataset.

5.3 Event Data Annotation Quality Evaluation

Scenario slot annotation performance has been evaluated on both the development and test sets using the standard Precision and Recall metrics. A sample annotation of Event_Type, Event_Reason and Event_Location slots in the multilingual development set is given in Fig. 5.1.

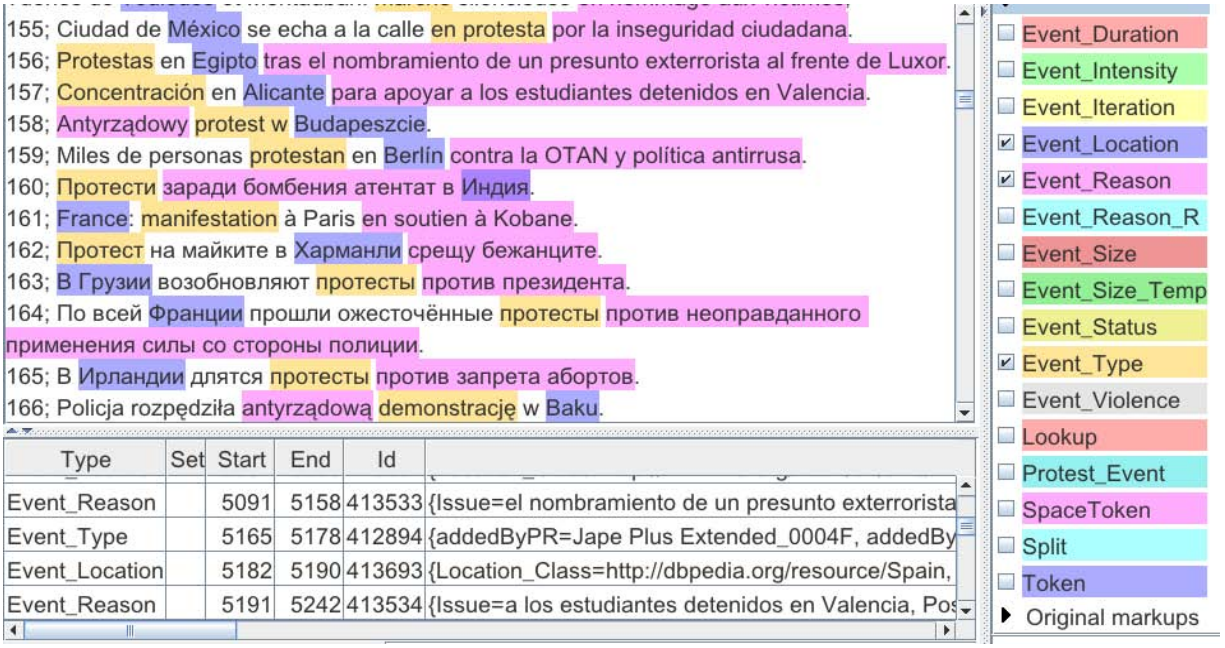

FIGURE 5.1: A sample annotation of Event_Type, Event_Reason and Event_Location in GATE GUI

The annotation quality has been calculated as follows. The development and test sets have been divided into language-specific subsets. The number of false positive, false negative and true positive annotations has been counted. An annotation is considered a true positive if it covers and does not lie beyond the natural language representation of a semantic component (for Event_Reason Issue feature and Event_Location) and is associated with a correct class/instance in the concept hierarchy (Event_Type, Event_Location, Event_Weight, Event_Reason Position feature). An annotation is considered a false positive if it does not cover or lies beyond the natural language representation of a semantic component. In case an extra function word is spanned by a given Event_Type or Event_Reason annotation, it will nonetheless be considered as a true positive. A false negative is a span of text corresponding to the natural language representation of a target semantic component that has not been covered by the target annotation. The annotation evaluation has been accomplished by one multilingual expert.

5.3.1 Evaluation Metrics

Precision is the fraction of retrieved documents that are relevant, and Recall is the fraction of relevant documents that are retrieved[1]:

[1] http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-unranked-retrieval-sets-1.html

$$Precision = \frac{|G \bigcap C|}{|G|}, Recall = \frac{|G \bigcap C|}{|C|}, \tag{5.1}$$

where G is the total amount of sequences retrieved from the reports' lead sentences for a given scenario slot, and C is the amount of documents that contain text spans, which are approved by the expert as relevant representatives of the same slot.

5.3.2 Results

In Tables 5.3 and 5.4 we present the results of annotation quality evaluation. Here, ET stands for Event_Type, ER - Event_Reason, EL - Event_Location, EW - Event_Weight.

TABLE 5.3: Annotation quality evaluation on the development set

Language	**No. of Docs**	**No. of Annots**	**Precision**				**Recall**			
			ET	**ER**	**EL**	**EW**	**ET**	**ER**	**EL**	**EW**
Bulgarian	93	244	1	0.97	0.93	1	1	0.97	0.82	1
French	85	218	1	0.94	0.92	0.98	1	0.98	0.87	0.88
Polish	93	231	1	0.96	0.96	0.98	1	0.72	0.90	0.90
Russian	107	293	1	0.98	1	0.92	1	0.98	0.89	0.80
Spanish	89	218	1	1	0.83	0.98	1	0.94	0.78	1
Swedish	108	195	1	0.98	1	1	1	0.95	0.65	0.88

TABLE 5.4: Annotation quality evaluation on the test set (500 documents per language)

Language	**No. of Annots**	**Precision**				**Recall**			
		ET	**ER**	**EL**	**EW**	**ET**	**ER**	**EL**	**EW**
Bulgarian	1053	0.99	0.98	0.95	0.87	1	0.97	0.87	0.86
French	1093	0.92	0.92	0.91	0.66	0.99	0.90	0.91	0.62
Polish	893	0.91	0.96	0.90	0.90	1	0.84	0.84	0.67
Russian	1031	0.93	0.95	0.97	0.67	0.99	0.90	0.89	0.75
Spanish	1023	0.96	0.93	0.90	0.90	0.99	0.88	0.93	0.83
Swedish	906	0.99	0.97	0.95	0.80	1	0.95	0.91	0.54

Chapter 6

Conclusion

This Chapter recalls the main objective and tasks of the work, describes its original contributions, discusses evaluation results and outlines future steps.

The objective of this thesis is to enrich the protest event analysis unit (event scenario) with new relevant features that has not been covered by the previous systems (GDELT, W-ICEWS, SPEED, El:DIABLO, [55], [20]) and mitigate the problem of training data absence in multilingual protest event data collection and coding by using a knowledge-driven approach.

To this end, the following tasks have been accomplished:

- Corpus collection and filtering
  - a corpus of 14464 multilingual news lead sentences (from Bulgarian, French, Polish, Russian, Spanish, Swedish tabloids) has been collected with Scrapy 1.0;
  - subcorpus of 13710 lead sentences related to protest events has been formed using a Python 2.7 script;
- Data analysis, concept hierarchy construction, feature selection
  - protest event descriptions have been analyzed, and domain-specific concepts (protest events hierarchy, subevents and properties) have been structured and formalized in Protégé - 4.3;
  - out of the whole feature set, the following have been selected to be introduced in the current version:
    - Event_Type: the what of the event, e.g.: rally, march, boycott, strike, picketing, hunger strike, riot, symbolic act, etc..

    - Event_Location: the where of the event, e.g.: names of countries, cities and physical settings (DBpedia lookup).
    - Event_Reason: the why of the event or position of a protesting group towards an issue: for a cause (expressions of support incl. conmemorations and demands) or against a cause.
    - Event_Weight: an attribute that defines the importance of an event and takes into account the values of the following slots: Event_Duration, Event_Intensity, Event_Iteration, Event_Size, Violence_Use.

- Software selection for the efficient multilingual text processing
  - GATE 8.0 has been selected as the framework for multilingual text processing, automatic knowledge resources population and output generation;
  - the performance and feature sets of PoS taggers, the natural language analysis basis, have been studied; as a result, the Treetagger plugin (Tagger_Framework) has been selected, because it covers the majority of the considered languages, except Swedish, however, it does not provide rich feature sets for French and Spanish. The Stockholm tagger has been selected for Swedish texts processing, its output has been adapted using a Python 2.7 script;
  - DBpedia remote repository has been selected for the current version as the location knowledge base;

- Multilingual knowledge resources building
  - JAPE pattern/rule pairs have been built according to GATE 8.0 standards. The resulting multiphase grammar includes 12 phases. Patterns take into account the properties of noun phrase construction in each of the languages. First, the protest Event_Type is identified (verb, noun or adjective) and annotated with the corresponding ontology class using a single-pattern grammar. Next, the main descriptors are annotated: Event Reason grammar (7 patterns), and Event Location grammar (4 patterns). At the next stage, the rest of the concepts are highlighted in case they are present in a given sentence: Event Size (5 patterns), Event Duration (3 patterns), Event Iteration (3 patterns), Violence Involved (1 patterns), Event Intensity (1 pattern), and Event Status (1 pattern). Finally, the features of the mentioned annotations together with the Event ID are added to the main Protest Event annotation that covers the whole headline.
  - Gazetteers have been built according to GATE 8.0 standards. Each list of each gazetteer includes multilingual entries. The gazetteers' quantitative characteristics as follows:

Gazetteer name	No. of lists	No. of entries
ETGazetteer	20	252
ERGazetteer	10	225
EWGazetteer	15	364
AuxGazetteer	12	768
GeoGazetteer	6	8841

  - Eight SPARQL queries to the corresponding DBpedia location-related classes have been constructed;

- Corpus pipeline building A sequence of processing resources have been integrated into a corpus pipeline to perform annotation on the basis of the developed knowledge resources: Document normalizer, GATE Unicode Tokenizer, ANNIE Sentence Splitter, Large KB Gazetteer (DBpedia lookup), Generic Tagger, OntoGazetteer, Flexible Gazetteer, BWP Gazetteer, ANNIE Gazetteer, JAPE Plus Extended;

- System output setting

  - Configurable Exporter PR is incorporated into the pipeline at the final stage to output scenarios into CSV so that they can be further processed and visualized;

  - A JAPE grammar is built to facilitate ontology population with new instances;

- Performance of experiments on test sets The performance has been tuned on the development set and tested on both development and test sets.

The **original contributions** of this work include the creation of (i) a multilingual corpus of news reports related to protest events; (ii) knowledge resources for protest event feature extraction (concept hierarchy, gazetteers, grammars); (iii) a pipeline for the annotation and CSV output of protest event features.

6.1 Discussion of Results

6.1.1 Event Selection

Within the event selection task, the selection of protest-related headlines from news feeds using two lists of keywords (main and auxiliary) is performed. The main condition for the query is to contain keyword substrings from both lists (e.g., "protest" and "against" or "protest" and "in"). This query together with stop words removal ensures good results, however, some of the main keywords represented by such lexemes as "block", "march", "action", "manifestation" cause identical errors due to their ambiguous nature. In the

future work section we outline the steps to be taken to tackle the ambiguity issues. This simple approach to the event selection/topic identification problem has been undertaken, because its results are satisfactory for this case, and this is not the main focus of this study.

6.1.2 Event Features Annotation

Within the feature annotation task, the knowledge-based finite state approach has been undertaken to annotate protest events in the filtered dataset and thus perform the final selection of relevant events with their respective features. The algorithm is able to distinguish between different types of events (boycotts, marches, labour actions, etc.), protesters' negative and positive attitude towards some issues, their demands, the location of a given action (country/city/physical setting) and its weight. The weight is equal to "1" if the algorithm detects a massive event, a sequence of events, violence use, etc., otherwise it is "[null]".

The best performance on the test set for the considered languages has been achieved for the Event_Type slot with Precision and Recall between 0.91 and 1. The lowest Precision values (0.91 (Polish), 0.92 (French), 0.93 (Russian)) are due to the incompleteness of the stop words list and the use of a minimum edit distance gazetteer (BWP). For instance, in the Polish test set there are lots of events triggered by the keyword substring "blok", which refer to traffic accidents and can be filtered out using additional stopwords and query conditions. BWPGazetteer use resulted in false annotations, because the distance is calculated for the whole word, the application of the algorithm on a specific word part (beginning or ending) cannot be manipulated from the runtime settings. For instance, the French word "profession" has been wrongly attributed to the event type "procession".

The Precision between 0.92 and 0.98 has been achieved for the Event_Reason slot. The lowest values (0.92 (French), 0.93 (Spanish)) are due to the ambiguity of lexemes containing "pro" and "anti" substrings. Also, ambiguity issues arise, because subevents are not taken into account in the current implementation. For instance, in French and Spanish headlines reporting the cancellation subevent ("protest cancelled because of/by"), Event_Reason has been wrongly annotated as introduced with the preposition "pour/-por": "protest cancelada por" "protestation annulée pour". When denoting an actual event reason (protesters' position), the preposition "за/pour/за/por/för" ("for") also results ambiguous, because it means either "in support of" or just "related to". The Recall values for the given slot lie between 0.84 and 0.97. The lowest are 0.84 (Polish) and 0.88 (Spanish). The manual analysis of results allowed to find several expressions

that had not been introduced into the gazetteers of event reason before and lead to this performance.

Event location grammar performance is evaluated on the test set as follows. Precision values lie between 0.90 and 0.97, where the lowest ones are obtained for Polish and Spanish (0.90) and French (0.91). In the first place, this is due to false positive annotations of nationality adjectives (GeoGazetteer) and peoples names (DBpedia). Examples show that we cannot rely on these mentions without any decision support from the main article text. A significant number of false positives comes from the wrong annotation of company and person names as event location. Also, the fact that the "wholeWordsOnly" and "caseSensitive" options in BWPGazetteer are set to "false" leads to the wrong annotation of substrings as US, LA, etc.. The Recall is between 0.84 and 0.93 with the lowest values for Polish (0.84), Bulgarian (0.87) and Russian (0.89). The amount of false negatives for these languages is higher due to the following: (i) the insufficiency of DBpedia and GeoGazetteer data, (ii) no possibility of taking into account morphological variations of location names when using the Large KB Gazetteer to connect to DBpedia.

Event weight grammar performance is evaluated as follows. The Precision for the considered languages lies within the interval of 0.66-0.90, where the lowest values are obtain for the French (0.66) and Russian (0.67) languages. In this case, the performance is mainly influenced by the runtime settings of the BWPGazetteer ("wholeWordsOnly = false"). Also, according to the manual analysis of false positives, the presence of negation and phrases that deny the proposition evidencing a non-null event weight should be taken into account to discard irrelevant mentions. The Recall within 0.54-0.86 with the lowest values for Swedish (0.54), French (0.62) and Polish (0.67) is mainly due to the incompleteness of the corresponding gazetteers.

To sum up, the detailed examination of results has shown that lower Precision values are caused by ambiguity issues and BWPGazetteer use. The lower Recall is due to the incompleteness of knowledge resources and partially to the PoS tagging quality.

6.2 Future Work

The present Section proposes solutions to the previously introduced problems and brings in some ideas. Within the event selection task, several approaches should be tested to tackle the ambiguity issues: (i) introducing additional constraints to the main crawling script that include (a) considering full keywords (tokens) instead of substrings, and (b) taking the relative position of keywords (coming from the main and auxiliary lists) in a headline into account; (ii) applying a machine learning algorithm to group protest-related

headlines or full articles, where the simultaneous presence of two or more scenario slots is one of the input features. Within the event annotation task, the following steps should be undertaken to improve the performance of the algorithm and make its use more practical: (i) perform lemma or stem-based lookup of a multilingual geodata-oriented repository, such as GeoNames, (ii) retrain PoS taggers from the TreeTagger package on larger corpora or include multiple multilingual PoS taggers to be able to annotate relevant features, such as noun plurality, animacy, etc., for the whole set of languages, as well as to reduce the error rate (iii) use the Extended Gazetteer PR instead of BWP that allows for the adjustment of multiple runtime settings including matching gazetteer entries at the word beginning/ending, selection of longest matches, etc..

euroPEA pipeline is able to process around 500 sentences per 1 sec., which is faster than the pattern-based PETRARCH (150 sentences per 1 sec.). However, GATE annotation framework has very well-known drawbacks related to the runtime performance when processing heavy grammars and large corpora. If GATE does not meet the new needs related to future system improvements, one of the following approaches will be selected: (i) map knowledge resources into another framework with a higher processing capacity, (ii) train event feature classifiers. Also, in order to build topic clusters of reasons and capture the variety of event reason text representations, a classifier will be trained on the instances identified from the headlines and the underlying full reports. The articles will be clustered, according to their content similarity, and the produced reason clusters will be populated from the corpus, adding new textual reason representations for a given topic.

Although there are minor changes that will indeed contribute to the selection and annotation performance enhancement, the most important part of the future work consists in the collaboration with a multidisciplinary team of sociologists, data scientists and other researchers in order to find support for the system development and build a fully-fledged and efficient tool for protest event analysis.

Appendix A

Protest Event Concept Hierarchy

*maximum three-level deep, Event_Type hierarchy is included in the thesis body

1 Event_Type
1.1 Verbal_Expression
1.2 Protest_Action
1.2.1 Non-violent_resistance
1.2.1.1 protest
1.2.1.1.1 boycott
1.2.1.1.2 direct_action
1.2.1.1.2.1 street_protest
1.2.1.1.2.1.1 demonstration
1.2.1.1.2.1.2 casserole
1.2.1.1.2.1.3 concert
1.2.1.1.2.1.4 flash_mob
1.2.1.1.2.1.5 escrache
1.2.1.1.2.1.6 blockade
1.2.1.1.2.1.6.1 picket
1.2.1.1.2.1.6.2 sit-in
1.2.1.1.2.1.7 march
1.2.1.1.2.1.7.1 torchlight_procession
1.2.1.1.2.1.7.2 pride_parade
1.2.1.1.2.1.7.3 silent_protest
1.2.1.1.3 symbolic_acts
1.2.1.1.3.1 hunger_strike
1.2.1.1.3.2 resignation
1.2.1.1.4 job_action

1.2.1.1.4.1 work-in

1.2.1.1.4.2 business_protest

1.2.1.1.4.3 slowdown

1.2.1.1.4.4 work-to-rule

1.2.1.1.4.5 strike

1.2.1.1.4.5.1 general_strike

1.2.1.1.4.5.2 sit-down

1.2.1.1.4.5.3 sympathy_strike

1.2.1.1.4.5.4 walkout

1.2.1.2 insurgency

1.2.2 Violent_attacks

1.2.2.1 sabotage

1.2.2.2 riot

1.2.2.2.1 race_riot

1.2.2.2.2 food_riot

1.2.2.3 mutiny

1.2.2.4 uprising

1.2.2.4.1 spontaneous

1.2.2.4.2 organized

1.2.2.5 revolution

1.2.2.6 Coup_d_État

2 Event_Property

2.1 Event_Actor

2.1.1 Authority

2.1.2 Named_Group

2.1.3 Organization

2.1.4 Person

2.2 Event_Date

2.3 Event_SCALE

2.3.1 Duration

2.3.2 Intensity

2.3.3 Iteration

2.3.4 Size

2.3.5 Violence_Involved

2.4 Event_Location

2.4.1 Africa

2.4.2 America

2.4.3 Asia

2.4.4 Europe

2.4.5 Oceania
2.5 Event_Reason
2.5.1 Issue
2.5.2 Position
2.6 Event_Status
2.6.1 Finished
2.6.2 In_Progress
2.6.3 Never_Took_Place
2.6.4 Planned
3 Antecedent_Type
3.1 Appeal
3.2 Warn_on_Disruption
3.3 Authorize
3.4 Ban
3.5 Cancel
3.6 Request
3.7 Threat
3.8 Change
4 Aftermath_Type
4.1 Damage
4.1.1 Economic_Damage
4.1.2 Other
4.2 Response
4.2.1 Ignore
4.2.2 Negative
4.2.3 Positive
4.3 Violence
4.3.1 Brawl
4.3.2 Clash_with_Authority
4.3.3 Property_Capture
4.4 Event

Appendix B

Macro Patterns

```
Macro: PREMOD

(
({Token.category == ART}|{Token.category == PDEL}|{Token.category ==
    PAL}|{Token.category == "DET:ART"})?
(
{Token.category == ADJ}|{Token.category == VLadj}|{Token.category =~
    "adj"}|
{Token.category =~ "ppas"}|{Token.category =~ "pact"}|{Token.category =~
    "Vpp"}|{Token.category =~ "A"}|{Token.category =~ "Vmp"}
)[1,2]
)

Macro: POSTMOD

(
(
{Token.category == ADJ, !Lookup}|
{Token.category == VLadj, !Lookup}|
{Token.category =~ "adj", !Lookup}|
{Token.category =~ "ppas", !Lookup}|
{Token.category =~ "pact", !Lookup}|
{Token.category =~ "Vpp", !Lookup}|
{Token.category =~ "A", !Lookup}|
{Token.category =~ "Vmp", !Lookup}
)[1,2]
)

Macro: HEAD

({Token.category =~ "N|subst"})
```

```
Macro: NG

(
(PREMOD)?
(HEAD)
(POSTMOD)?
)

Macro: ANTI

(
{
Token.string =~ "Аанти([])|([Aa]nt[iy])",
Token.string !~ "([Aa]nti[gq])Иинфанти|([])|([Ii]nfanti)",
Token.length > 4
}
)

Macro: PRO

(
{
Token.string =~ "Ппро([])|([Pp]ro)",
Token.string !~ "([Pp]ro[vw]o[ck])Ппродоълж|([][])Ппрово|([])|
([Pp]rotest)Ппротест|([])против|()", Token.length > 3, Token.category !~
    "Vmi[spf]"
}
)

Macro: BASE

(
{Token.kind == word}
(
{Token.kind != punctuation,
Token.string !=~ "sece|",
Token.string !=~ "s'",
Token.category != "VLfin",
Token.category != "VER:pres",
Token.category !~ "Vp[ip][it]f",
Token.category !~ "praet",
Token.category !~ "fin",
Token.category !~ "Vmi[spf]"}
)[0,12]
)

Macro: CONTENT
```

```
(
(BASE)
(
 (({Token.string == ","})(BASE))[0,3]
 (({Lookup.minorType == and})(BASE))
)?
)

Macro: EVENT_TYPE

(
({Event_Location}|{Event_Reason})[0,2]
(PREMOD)?
{Event_Type}
(POSTMOD)?
({Event_Location}|{Event_Reason})[0,2]
)

Macro: DURATION

(
(
({Lookup.majorType == Numeral}|{Token.kind == number})
({Token.length < 4})[0,2]
{Lookup.majorType == Date_unit}
)
|
(
{Lookup.majorType == Date_unit}
({Lookup.majorType == Numeral}|{Token.kind == number})
({Token.length < 4})[0,2]
)
)
```

Appendix C

Crawler base code

```
# Set the default encoding to utf-8
# -*- encoding: utf-8 -*-

"""
Console program for keyword-based text search on the web.
The present module requires the installation of Python 2.7
    (http://python.org) and Scrapy web crawling framework
    (http://scrapy.org)
The code is partially provided by the freeshare source
http://opensourcehacker.com/2011/03/08/installing-and
-using-scrapy-web-crawler-to-search-text-on-multiple-sites/
The original indentation should be preserved.
"""

# Import Scrapy libraries
import StringIO
from functools import partial
from scrapy.http import Request
from scrapy.spider import BaseSpider
from scrapy.contrib.spiders import CrawlSpider, Rule
from scrapy.contrib.linkextractors.sgml import SgmlLinkExtractor
from scrapy.selector import Selector
from scrapy.item import Item


# Extract all substrings for a given string (keyword)
def find_all_substrings(string, sub):
    """

http://code.activestate.com/recipes/499314-find-all-indices-of-a-substring
-in-a-given-string/
```

```
    """
    import re
    starts = [match.start() for match in re.finditer(re.escape(sub),
    string)]
    return starts

# Create a spider
class MySpider(CrawlSpider):
    # Define the name for the command line use
    name = "RussianNews"
    # Define the allowed domains and start urls.
    # The following lists of news sources were created for the purposes of
    socio-political event extraction from Russian news.
    allowed_domains = [
        "ikd.ru/",
        "aftershock.su",
        "bbc.co.uk/russian",
        # ...
    ]

    start_urls = [
        "http://ikd.ru/",
        "http://aftershock.su",
        "http://bbc.co.uk/russian",
        # ...
    ]

    # Set the callback function for every link being found
    rules = [
        Rule(SgmlLinkExtractor(), follow=True, callback="check_key_words")
    ]

    # Count the number of pages crawled
    crawl_count = 0

    # Define the callback function that checks the presence of keywords at
    each web page crawled
   print "["
    def check_key_words(self, response):

        # Report the count of web pages
        self.__class__.crawl_count += 1

        crawl_count = self.__class__.crawl_count
        if crawl_count % 1000 == 0:
            print "%d pages crawled" % crawl_count

        # Create the lists of keyword substrings (case-sensitive).
```

```
    # Main keyword substrings (mainkw) define the main concept being
searched. In the present case - different names of protest events.
    # Other keyword substrings (otherkw) define the related concept,
which should accompany the main event in text. In our case - the
origin of protest.
    mainkw = [
        "митинг",
        " акци",
        # ...
    ]

    otherkw = [
        "в поддержку",
        "в честь",
        # ...
    ]
    # Define the list of stopword substrings in order to further
exclude unrelated texts
    stopw=[
    "турмаршрут",
    "путешестви",
    # ...
    ]
    contextkw = [
    "Москв",
    "Госдум",
    "РФ",
    # ...
    ]

    # Extract response URL
    url = response.url

    # Check response content type. PDFs are omitted.
    ct = response.headers.get("content-type", "").lower()
    if "pdf" in ct:
        return Item()
    else:
        data = response.body

    # Define selector for further extraction
    sel = Selector(response)

    # Extract title and, if found, encode it as utf-8
    title = sel.xpath("//title/text()").extract()
    if len(title) > 0:
        title = title[0].encode('utf-8')
    else:
```

```
            title = ''

        # Extract subtitle and, if found, encode it as utf-8
        subtitle=sel.xpath('//meta[contains(@name,
    "description")]/@content').extract()
        if len(subtitle) > 0:
            subtitle = subtitle[0].encode('utf-8')
        else:
            subtitle = ''

        # Extract metadata containing news article keywords
        metainfo=sel.xpath('//meta[contains(@name,
    "keywords")]/@content').extract()

        # Extract all header and article tags in order to further extract
    data for the main article (date and time)
        headers = sel.xpath('//header')
        articles = sel.xpath('//article')

        # Define other date/time extractors
        itartass_time =
    sel.xpath('//span[@class="b-material__date"]/text()').extract()
        interfax_time =
    sel.xpath('//meta[contains(@property,"article:published_time")]
/@content').extract()
        time=sel.xpath('//time[contains(@class,
    "b-article__publish_date")]/@datetime').extract()
        tria=sel.xpath('//time[@class="article_header_date"]
/@datetime').extract()

        # Define article body extractor and encoder
        body = sel.xpath("//p/text()").extract()
        encodedBody=[]
        for p in body:
            p=str(p.encode("utf-8"))
            encodedBody.append(p)
        stringBody='\n'.join(map(str, encodedBody))

        # Check the presence of main keyword substrings in title. Proceed
    if at least one keyword is present.
        for kw1 in mainkw:
            found = False
            mainkw_substrings = find_all_substrings(title, kw1)
            if len(mainkw_substrings) > 0:
                for kw2 in otherkw:
                    otherkw_substrings = find_all_substrings(title, kw2)
                    if len(otherkw_substrings) > 0:
                        for kw3 in contextkw:
```

```
                            contextkw_substrings =
    find_all_substrings(stringBody, kw3)
                            if len(contextkw_substrings) > 2:
                                found = True
     print "{"
                print "\"URL\":\"%s\"" % url + ","
                print "\"Title\":\""+title+ "\""+ ","
     # print subtitle and metadata, if any
                                print "\"Subtitle\":\""+subtitle+ "\""+ ","
                                for w in metainfo:
                                    w=w.encode('utf-8')
                                    print "\"Metadata\":\""+w+ "\""+ ","
                                print "\"Text body\":\""+stringBody+ "\""+
    ","
     # print time
                                for time in headers.xpath('time/text()'):
                                    print time.extract()
                                for fdtime in
    articles.xpath('//div[@class="feed-item-date"]/text()'):
                                    print "\"Time\":\""+fdtime.extract()+
    "\""+ ","
                                print
    "\"Itartass_Time\":\""+itartass_time+ "\""+ ","
                                print
    "\"Interfax_Time\":\""+interfax_time+ "\""+ ","
                                print "\"Time\":\""+time+ "\""+ ","
                                print "\"RiaNovosti_Time\":\""+tria+ "\""
                print "},"

            if found is True:
                break
        print "]"
        return Item()

# Define the type of requests to follow.
# Avoid links in binary data.
    def _requests_to_follow(self, response):

        if getattr(response, "encoding", None) != None:

                return CrawlSpider._requests_to_follow(self, response)
        else:
                return []
```

Appendix D

Crawler settings

Language|Parameters	Crawling domains	Trigger terms	Co-occurring concepts
Bulgarian	"http://abc.bg", "http://novini.bg", "http://dnes.bg", "http://btvnovinite.bg", "http://dnevnik.bg", "http://novinite.bg", "http://vesti.bg", "http://24chasa.bg", "http://actualno.bg", "http://bnews.bg/bulgaria", "http://bulgaria-news.bg", "http://novinibg.com"	"Протест", "протест", "Шестви", "шестви", "Митинг", "митинг", "Демонстраци", "демонстраци", "Манифестаци", "манифестаци", "Стачк", "стачк", "Бунт", "бунт", "Марш", "марш", "Бойкот", "бойкот", "пикетира"	"с искания за", "против", "срещу", "Анти", "анти", "в знак за", "заради", "в подкрепа на", "в памет на", "по повод", "планира", "при", "обяви", "на протест", "След", "след"

Language|Parameters	Crawling domains	Trigger terms	Co-occurring concepts
French	"lemonde.fr", "france24.com", "rfi.fr", "ledevoir.com", "ndf.fr", "lanouvellerepublique.fr", "rtl.fr", "aujourd-hui.ma", "france3-regions.francetvinfo.fr", "franceinfo.fr"	"Démonstration", "démonstration", "Boycott", "boycott", "croisade", "Grève", "grève", "Marche", "marche", "Révolte", "révolte", "Mobilisation", "mobilisation", "Protestation", "protestation", "Manifestation", "manifestation", "Parade", "parade", "Contestation", "contestation", "Rassemblement", "rassemblement"	"contre", "pour", "en soutien à", "anti", "Anti", "à la gloire", "pour glorifier", "en hommage", "en mémoire de", "à", "au", "Au" "en", "En", "mouvement", "blocage"

Language|Parameters	Crawling domains	Trigger terms	Co-occurring concepts
Polish	"http://wiadomosci.wp.pl", "http://supernowosci24.pl", "http://nowosci.com.pl", "http://tvn24.pl", "http://wiadomosci.onet.pl", "http://wiadomosci.gazeta.pl", "http://polska.newsweek.pl", "http://dziennik.pl", "http://se.pl/wydarzenia/", "http://polskieradio.pl", "http://fakt.pl", "http://fakty.interia.pl"	"Protest", "protest", "Strajk" "strajk", "Manifestacj", "manifestacj", "Demonstracj", "demonstracj", "demonstrant", "Pikiet", "pikiet", "Blokada", "blokada", "blokow", "blokuj"	" przeciwko ", "sprzeciwiają się", " przeciw ", " przez ", " w ", " pod ", " z ", "tys", " obok ", " na ", " W ", "aktywist"

Language|Parameters	Crawling domains	Trigger terms	Co-occurring concepts
Russian	"ikd.ru/", "aftershock.su", "bbc.co.uk/russian", "itar-tass.com", "rusplt.ru", "bfm.ru", "russian.rt.com", "slon.ru", "kurs.ru", "news.megatyumen.ru", "echo.msk.ru", "rus-novosti.ru", "15minut.org", "govoritmoskva.ru", "amurburg.ru", "ria.ru", "lenta.ru", "fontanka.ru", "forbes.ru", "livejournal.com", "gazeta.ru", "rbc.ru", "news.rambler.ru", "vesti.ru", "newsru.com", "news.yandex.ru", "inter-fax.ru", "news.bigmir.net", "kommersant.ru", "hope-sandfears.com/news", "kp.ru", "mk.ru", "ng.ru", "gazeta.ua/ru/", "pravda.ru", "rg.ru", "trud.ru"	"Митинг", "митинг", "Акци" " акци", "выступ", "Протест", "протест", "Шестви" " шестви", "Марш", "марш", "митинг-концерт", "Демонстраци" "демонстраци", "ЛГБТ-митинг", "Гей-парад" "гей-парад", "Парад", "парад", "бастов", "басту", "Бунт" "бунт", "Голодовк" "голодовк", "Пикет", "пикет", "Восстани", "восстани", "Беспорядк", "беспорядк", "Бойкот", "бойкот"	"в поддержку", "в честь", "против", "по поводу", "с требованием", "в защиту", "по событиям", "в ожидании", "памяти", "по случаю", " за ", "из-за", "сторонник", "анти", "Анти", "требуя"

Language|Parameters	Crawling domains	Trigger terms	Co-occurring concepts
Spanish	"http://elmundo.es", "http://elpais.com", "http://rtve.es", "http://elconfidencial.com/espana", "http://abc.es", "http://antena3.com", "http://20minutos.es", "http://que.es", "http://lavanguardia.com", "http://elperiodico.com", "http://lasprovincias.es", "http://laopinion.es", "http://actualidad.rt.com", "http://ragap.es/actualidad", "http://publico.es/espana", "http://libertaddigital.com"	"Protesta", "protesta", "Marcha", "marcha", "Manifestaci", "manifestaci", "Manifiest", "manifiest", "Huelg", "huelg", "Desfile", "desfile", "Movimiento social", "movimiento social", "Escrache", "escrache", "Concentraci" "concentraci", "movimientos ciudadanos", "Movimientos ciudadanos"	"por", "contra", "en defensa de", "en apoyo a", "de apoyo", "para apoyar", "reclama" "en homenaje a"

Language|Parameters	Crawling domains	Trigger terms	Co-occurring concepts
Swedish	"aftonbladet.se", "vt.se", "corren.se", "nt.se", "nsd.se", "norran.se", "sac.se", "nyheter24.se", "svt.se", "dn.se", "ex-pressen.se", "sveriges-radio.se", "metro.se", "dagen.se", "op.se", "in-ternationalen.se", "sven-ska.yle.fi"	"Demonstration", "demon-stration", "demonstrera", "Bojkott", "bojkott", "Protest", "protest", "Man-ifestation", "manifestation" "manifestera", "Strejk", "strejk", "Marsch", "marsch", "blockeras", "Kravall", "kravall", "Up-plopp", "upplopp", "Up-pror", "uppror", "Myteri", "myteri", "Procession", "procession"	" för ", " mot ", " till följd av ", " i ", "till minne av", "till stöd för", "till försvar för", "för att hedra"

Appendix E

Freeling PoS tagger output converter to GATE XML

```
// Copyright 2015, yarnaid
// Based on the sample.cc code

#include <iostream>
#include <string>
#include <list>

#include "freeling.h"
#include "freeling/morfo/traces.h"

// using namespace std;
using std::string;
using std::list;
using namespace freeling;

// predeclarations
void PrintMorfo(list<sentence> &ls);
wstring StringToWstring(const string &source);
int tag_counter = 1;


int main(int argc, char **argv) {
    wstring text;
    list<word> lw;
    list<sentence> ls;

    wcout << L"<?xml version=\"1.0\" encoding=\"utf-8\"?>" << endl;

    /// set locale to an UTF8 comaptible locale
    util::init_locale(L"default");
```

```
wstring ipath;
// TODO(yarnaid): replace with getopt and add other options to use
below
if (argc < 2) {
    ipath = L"/usr/local";
} else {
    ipath = util::string2wstring(argv[1]);
}

wstring path = ipath + L"/share/freeling/es/";

// create analyzers
tokenizer tk(path + L"tokenizer.dat");
splitter sp(path + L"splitter.dat");
maco_options opt(L"es");
opt.UserMap = false;
opt.QuantitiesDetection = false; //deactivate
ratio/currency/magnitudes detection
opt.AffixAnalysis = true;
opt.MultiwordsDetection = true;
opt.NumbersDetection = true;
opt.PunctuationDetection = true;

opt.DatesDetection = true;
opt.QuantitiesDetection = false;
opt.DictionarySearch = true;
opt.ProbabilityAssignment = true;
opt.NERecognition = true;
opt.UserMapFile = L"";
opt.LocutionsFile = path + L"locucions.dat";
opt.AffixFile = path + L"afixos.dat";
opt.ProbabilityFile = path + L"probabilitats.dat";
opt.DictionaryFile = path + L"dicc.src";
opt.NPdataFile = path + L"np.dat";
opt.PunctuationFile = path + L"../common/punct.dat";
// create the analyzer with the just build set of maco_options
maco morfo(opt);
// create a hmm tagger for spanish (with retokenization ability, and
forced
// to choose only one tag per word)
hmm_tagger tagger(path + L"tagger.dat", true, FORCE_TAGGER);
// create chunker
chart_parser parser(path + L"chunker/grammar-chunk.dat");
// create dependency parser
dep_txala dep(path + L"dep/dependences.dat",
parser.get_start_symbol());
// get plain text input lines while not EOF.
```

```
    wcout << L"<paragraph gate:annotMaxId=\"2122001999\"  " \
          << "gate:gateId=\"" \
          << 0 \
          << L"\" xmlns:gate=\"http://www.gate.ac.uk\" >" \
          << endl;
    while (getline(wcin, text)) {
        // tokenize input line into a list of words
        lw = tk.tokenize(text);

        // accumulate list of words in splitter buffer, returning a list of
        // sentences. The resulting list of sentences may be empty if the
        // splitter has still not enough evidence to decide that a complete
        // sentence has been found. The list may contain more than one
        // sentence (since a single input line may consist of several
        // complete sentences).
        ls = sp.split(lw, false);

        // perform and output morphosyntactic analysis and disambiguation
        morfo.analyze(ls);
        tagger.analyze(ls);
        PrintMorfo(ls);

        // clear temporary lists;
        lw.clear();
        ls.clear();
    }

    // No more lines to read. Make sure the splitter doesn't retain
    anything
    sp.split(lw, true, ls);

    // analyze sentence(s) which might be lingering in the buffer, if any.
    morfo.analyze(ls);
    tagger.analyze(ls);
    PrintMorfo(ls);

    // parser.analyze(ls);
    // dep.analyze(ls);

    // close tag
    wcout << L"</paragraph>" << endl;
}


//--------------------------------------------
// print morphological information
//--------------------------------------------
```

```
void PrintMorfo(list<sentence> &ls) {
    word::const_iterator a;
    sentence::const_iterator w;
    // print sentence XML tag
    for (list<sentence>::iterator is = ls.begin(); is != ls.end(); is++) {
        // for each word in sentence
        for (w = is->begin(); w != is->end(); w++) {
            // for each possible analysis in word, output lemma, tag and
            // probability
            for (a = w->analysis_begin(); a != w->analysis_end(); ++a) {
                // print analysis info
                wstring lm = a->get_lemma();
                if (lm.compare(StringToWstring(string("\""))) == 0) {
                    lm = StringToWstring(string(""));
                }
                wcout << L"    <Morph lemma=\"" << lm;
                wcout << L"\" pos=\"" << a->get_tag();
                wcout << L"\" prob=\"" << a->get_prob();
                wcout << L"\" gate:gateId=\"" << tag_counter++;
                wcout << L"\">";
                wstring f = w->get_form();
                if (f.compare(StringToWstring(string("\""))) != 0) {
                    wcout << w->get_form();
                }
                break;
            }
            wcout << L"</Morph>";
        }
    }
}

wstring StringToWstring(const string &source) {
    wstring target(source.size() + 1, L' ');
    size_t newLength = mbstowcs(&target[0], source.c_str(), target.size());
    target.resize(newLength);
    return target;
}
```

Appendix F

PoS tagger output converter to GATE XML

* Converter of a PoS tagger (e.g., Stagger) output to GATE XML (Python)

```
# -*- coding: utf-8 -*-
# courtesy of yarnaid

import xml.etree.cElementTree as ET
from codecs import open

gate_id = 1
id_tag = 'gate:gateId'


root = ET.Element('paragraph')
root.set(id_tag, str(gate_id))
gate_id += 1
root.set('xmlns:gate', 'http://www.gate.ac.uk')
root.set('gate:annotMaxId', '2122001999')

# read data
items = []
with open('./swePOS.txt', 'r', 'utf-8') as f:
    for line in f.readlines():
        splited = line.split()
        if len(splited) == 2:
            word, tags = splited[0], splited[1]
            item = {
                'word': word,
                'tags': tags,
            }
            items.append(item)
```

```
for item in items:
    current_root = root
    element = ET.SubElement(current_root, 'Morph')
    element.set('lemma', item['word'])
    element.set('pos', item['tags'])
    element.set(id_tag, str(gate_id))
    gate_id += 1
    element.text = item['word']
    current_root = element

ET.ElementTree(root).write('out.xml', encoding='utf-8',
    xml_declaration=True)
```

Appendix G

Previously published work

1. Danilova V.: Cross-Language Plagiarism Detection Methods. In: Proc. of the Workshop on the Recent Advances in Natural Language Processing (RANLP), pp. 51-57, ISSN 1314-9156 (2013)

2. Popova S., Danilova V.: Document Representation for Clustering of Scientific Abstracts. In: Scientific and Technical Journal of Information Technologies, Mechanics and Optics, UDK 004-912, ISSN: 2226-1494, 1 (89) (included in the list of the Higher Attestation Commission of Russia) (2013)

3. Alexandrov M., Danilova V., Makarov A.: Document Representation Model for Internet Sociology Studies. In: "Mathematical modelling of social processes", Faculty of Sociology, Moscow State University, Collected Papers, Vol. 16, pp. 2-14 (2014)

4. Danilova V., Alexandrov M., Blanco X.: A Survey of Multilingual Event Extraction from Text. In: Proc. of the 19th Intern. Conf. on Application of Natural Language to Inform. Systems (NLDB-2014), pp. 85-88 (2014)

5. Koshulko O., Alexandrov M., Danilova V.: Forecasting Euro/Dollar Rate with Forex News. In: Proc. of the 19th Intern. Conf. on Application of Natural Language to Inform. Systems (NLDB-2014), pp. 148-153 (2014)

6. Popova S., Danilova V., Egorov A.: Clustering Narrow-Domain Short Texts using K-means, Linguistic Patterns and LSI. In: Proc. of the workshop on Analysis of Images, Social Networks, and Texts (AIST'2014). Springer LNCS, pp. 183-189 (2014)

7. Danilova V., Popova S.: Socio-Political Event Extraction Using a Rule-Based Approach. In: Proc. of the 13th International Conference on Ontologies, DataBases

and Applications of Semantics (ODBASE'2014), Springer, Volume 8842, 2014, pp. 537-546 (2014)

8. Danilova V.: Ontology Building and Annotation of Destabilizing Events in News Feeds. In: Artif. Intell. Applications to Business and Engineering Domains (Monograph), ITHEA Publ., pp.137-146 (2014)

9. Popova, S., Egorov, A., Khodyrev, I., Danilova, V.: Topic structure analysis of LiveJournal post groups related to the "Migration" domain. In: Mathematical modeling of social processes, MSU Faculty of Sociology, V. 17, pp. 155-177 (2015)

10. Alexandrov, M., Danilova, V., Blanco, X.: A modified tripartite model for document representation in Internet Sociology. In: Proceedings of DEXA'15 conference, part II, pp., 323-331, Springer LNCS (2015)

11. Danilova, V.: A Pipeline for Multilingual Protest Event Selection and Annotation. In: Proceedings of TIR'15 Workshop (Text-based Information Retrieval), pp.309-314, IEEE (2015)

Bibliography

[1] Agerri, R., Aldabe, I., Beloki, Z., Laparra, E., Lopez de Lacalle, M., Rigau, G., Soroa, A., van Erp, M., Vossen, P., Girardi, C. and Tonelli, S.: Event Detection. In: NewsReader. version 1. Deliverable D 4.2.1. Version FINAL (2013)

[2] Aleksić, V., Thurmair, G., Casamayor, G., Moumtzidou, A., Vrochidis, S.: D2.2. Basic techniques for speech recognition, text analysis and concept detection. MULTISENSOR Mining and Understanding of multilinguaL contenT for Intelligent Sentiment Enriched coNtext and Social Oriented inteRpretation, FP7-610411 (2014)

[3] Aone, C., Blejer, H., Okurowski, M.E., Van Ess-Dykema, C.: A Hybrid Approach to Multilingual Text Processing: Information Extraction and Machine Translation. In: Proceedings of the First Conference of the AMTA (1994)

[4] Arnulpy, B.: A Weighted Lexicon of French Event Names. In: Proceedings of the Second Student Research Workshop associated with RANLP 2011, pp. 9-16 (2011)

[5] Atkinson, M., Piskorski, J., Van der Goot, E., Yangarber, Y.: Multilingual Real-time Event Extraction for Border Security Intelligence Gathering. In: U.K. Wiil (ed.), Counterterrorism and Open Source Intelligence, Lecture Notes in Social 355 Networks 2, Springer-Verlag/Wien. (2011)

[6] Boschee, E., Weischedel, R., and Zamanian, A.: Automatic information extraction. In: Proceedings of the International Conference on Intelligence Analysis (2005)

[7] Anthony Boyles's blog: http://aaboyles.com/gdelts-unit-of-analysis-problem/

[8] Buitelaar, P., Netter, K., Xu, F.: Integrating Different Strategies for Cross-Language Information Retrieval in the MIETTA Project. In: Proceedings of the 14th Twente Workshop on Language Technology. (1998)

[9] Media Data and Social Science Research: Problems and Approaches. Cline Center for Democracy University of Illinois at Urbana-Champaign, February (2014)

[10] Danilova, V.: Cross-Language Plagiarism Detection Methods. In: Proceedings of the Student Research Workshop associated with RANLP 2013, pages 5157, Hissar, Bulgaria, 9-11 September (2013)

[11] Danilova V.: Ontology Building and Annotation of Destabilizing Events in News Feeds. In: Artif. Intell. Applications to Business and Engineering Domains (Monograph), ITHEA Publ., pp.137-146 (2014)

[12] Danilova V., Popova S.: Socio-Political Event Extraction Using a Rule-Based Approach. In: Proc. of the 13th International Conference on Ontologies, DataBases and Applications of Semantics (ODBASE'2014), Springer, Volume 8842, 2014, pp. 537-546 (2014)

[13] Dobrovol'skiy, D., Pöppel, L.: Lexical Synonymy within the Semantic Field POWER. In: Current Studies in Slavic Linguistics, ed. by Irina Kor Chahine, pp. 281 - 295 (2013)

[14] Du, M., Etter, P., Kopotev, M., Novikov, M., Tarbeeva, N., Yangarber, R.: Building Support Tools for Russian-Language Information Extraction. In: I. Habernal and V. Matousek (Eds.): TSD 2011, LNAI 6836, pp. 380387 (2011)

[15] Casals Elvira, X., Hecking, M.. IEPS: A Framework to Manage and to Visualize Information Extraction Results. Forschungsgesellschaft für Angewandte Naturwissenschaft e.V . (FGAN), Technischer Bericht FKIE/ITF/2005/2, September 2005.

[16] Fortuna's Corner blog: Why Big Data Mining Missed the Early Warning Signs of Ebola; over-reliance at the alter of big data mining will have some unpleasant shortfalls: http://fortunascorner.com/2014/09/29/why-big-data-mining-missed-the-early-warning-signs-of-ebola-over-reliance-at-the-alter-of-big-data-mining-will-have-some-unpleasant-shortfalls/

[17] Grishman, R.: Information Extraction: Capabilities and Challenges. In: Notes for the 2012 International Winter School in Language and Speech Technologies. Rovira i Virgili University. Tarragona, Spain (2012)

[18] R. Grishman, S. Huttunen, and R. Yangarber: Information Extraction for Enhanced Access to Disease Outbreak Reports. J. of Biomed. Informatics, 35(4) (2003)

[19] Halliday, M. A. K. : Halliday's Introduction of Functional Grammar. Routledge, Taylor & Francis Group, 4th edition, pp. 211-426 (2014)

[20] Hanna, A.: Developing a System for the Automated Coding of Protest Event Data - Available at SSRN 2425232 (2014)

[21] Harris, Z.: Papers in Structural and Transformational Linguistics, Dordrecht/ Holland: D. Reidel., x, 850 pp. (1970)

[22] Hayes, M., Nardulli, P. F.: SPEED's Societal Stability Protocol and the Study of Civil Unrest: an Overview and Comparison with Other Event Data Projects, October 2011, Cline Center for Democracy University of Illinois at Urbana-Champaign (2011)

[23] Hecking, M., Sarmina-Baneviciene, T.: A Tajik Extension of the Multilingual Information Extraction System ZENON. In: Washington, DC: CCRP C4ISR Cooperative Research Program, 14 pp. (2010)

[24] Hogenboom, F., Frasincar, F., Kaymak, U., de Jong, F.: An Overview of Event Extraction from Text. In: Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011), volume 779 of CEUR Workshop Proceedings, pages 48-57. (2011)

[25] Hornby, A., Oxford Advanced Learner's Dictionary of Current English. Oxford University Press, 28th Impression (2000)

[26] Keller, R. M.: "Classifiers, Acceptors, Transducers, and Sequencers". Computer Science: Abstraction to Implementation. Harvey Mudd College. p. 480. (2001)

[27] Leban, G., and Brank, J.: Early Event Extraction Prototype. Deliverable D 4.3.1, Version 1.0, (2011)

[28] Leetaru, K.: Automatic Document Categorization for Highly Nuanced Topics in Massive-Scale Document Collections: The SPEED BIN Program. Cline Center for Democracy University of Illinois at Urbana-Champaign, March (2011)

[29] Lejeune, G.: Structure Patterns in Information Extraction: a Multilingual Solution? In: Advances in Method of Information and Communication Technology AMICT09, Volume 11 p. 105-111, Petrozavodsk, Russia. (2009)

[30] Lieto, A.: Manual and semi-automatic domain-specific ontology building (master thesis), Università degli studi di Salerno (2008)

[31] McClelland, C.: World Event/Interaction Survey. ICPSR Inter-university Consortium for Political and Social Research, 1966-1978, ICPSR 5211, JANUARY (1999)

[32] Nardulli, P. F., Althaus, S. L., and Hayes, M.: A Progressive Supervised-learning Approach to Generating Rich Civil Strife Data (Forthcoming in Sociological Methodology)

[33] Naughton, M.: Sentence Level Event Detection and Coreference Resolution. PhD Thesis, National University of Ireland, Dublin (2009)

[34] Padró, L., Stanilovsky, E.: FreeLing 3.0: Towards Wider Multilinguality. Proceedings of the Language Resources and Evaluation Conference (LREC 2012), ELRA, Istanbul, Turkey (2012)

[35] Paleo B. W.: An Approximate Gazetteer for GATE based on Levenshtein Distance //Twelfth ESSLLI Student Session. – C. 197.

[36] Piskorski, J.: ExPRESS – Extraction Pattern Recognition Engine and Specification Suite. In: Proceedings of FSMNLP 2007 (2007)

[37] Piskorski, J., Belayeva, J., Atkinson, M.: Exploring the Usefulness of Cross-lingual Information Fusion for Refining Real-time News Event Extraction: A Preliminary Study. RANLP 2011: 210-217 (2011)

[38] Piskorski, J., Yangarber, R.: Information Extraction: Past, Present and Future. Survey Chapter in "Multi-source, Multilingual Information Extraction and Summarization", Theory and Applications of Natural Language Processing (T. Poibeau et al., eds.). Springer-Verlag, pp. 23-49 (2012)

[39] Pivovarova, L., Du, M., Yangarber, R.: Adapting the PULS Event Extraction Framework to Analyze Russian Text. At ACL: 4th Biennial Workshop on Balto-Slavic Natural Language Processing. Sofia, Bulgaria (2013)

[40] Pustejovsky, J., Knippen, R., Littman, J., Saurí, R.: Temporal and Event Information in Natural Language Text. In: Language Resources and Evaluation. 39 (2-3): 123-164 (2005)

[41] Raleigh, C., Linke, A., Hegre, H., and Karlsen, J.: "Introducing ACLED-Armed Conflict Location and Event Data." Journal of Peace Research 47, no. 5: 1-10 (2010)

[42] Schmid, H.: Improvements in Part-of-Speech Tagging with an Application to German. Proceedings of the ACL SIGDAT-Workshop. Dublin, Ireland (1995)

[43] Schrodt, P. A.: KEDS: Kansas Event Data System. Version 1.0. (1998)

[44] Schrodt, P. A.: CAMEO: Conflict and Mediation Event Observations. Event and Actor Codebook. Department of Political Science, Pennsylvania State University, Version: 1.1b3, March (2012)

[45] Schrodt, P. A.: GDELT: Global Data on Events, Location and Tone. Workshop at the Conflict Research Society, Essex University, 17 September (2013)

[46] Schrodt, P. A.: Open Event Data: Opportunities and Challenges. Workshop on "Big Data and Death", University of Wisconsin, 7 November (2014)

[47] Schrodt, P. A.: TABARI Textual Analysis by Augmented Replacement Instructions, Version 0.8.4, Parus Analytical Systems. Charlottesville, VA 22901, VERSION 0.8.4B3: August 30 (2014)

[48] Schrodt, P. A., Beieler, J., and Idris, M. : Three's a Charm?: Open Event Data Coding with EL:DIABLO, PETRARCH, and the Open Event Data Alliance. Presented at the International Studies Association, Toronto, March (2014)

[49] Schrodt, P. A. and Van Brackle, D.: Automated coding of political event data. In: Subrahmanian V. S. (ed.), Handbook of computational approaches to counterterrorism. Springer, New York, pp. 23-49 (2013)

[50] Schrodt, P. A., and Yonamine, J. : Automated Coding of Very Large Scale Political Event Data. Armed Conflict Location & Event Data Project (ACLED). Guide to Dataset Use for Humanitarian and Development Practitioners, pp. 3-9, January (2015)

[51] Steinberger, R.: A Survey of Methods to Ease the Development of Highly Multilingual Text Mining Applications. In: Language Resources & Evaluation 46: 155-176 (2012)

[52] Tadić, M.: Early Machine Translation Based Semantic Annotation Prototype. Deliverable D 3.3.1, Version 1.0 (2011)

[53] Toledo-Alvarado, J. I., Guzmaán-Arenas, A., Martiínez-Luna, G. L.: Automatic Building of an Ontology from a Corpus of Text Documents Using Data Mining Tools, Journal of Applied Research and Technology, Vol.10, No. 3, 398-404 (2012)

[54] Truswell, R.: Events, Phrases, and Questions. Oxford University Press, ISBN 978-0-19-957777-4 (Hbk.) (2011)

[55] Wueest,B., Rothenhäusler,K., Hutter,S.: Using Computational Linguistics to Enhance Protest Event Analysis - Available at SSRN 2286769 (2013)

[56] Wunderwald, M.,: Event Extraction from News Articles (Diploma Thesis), Dresden University of Technology. Dept. of Computer Science, (2011)

[57] Östling, R.: Stagger: an Open-Source Part of Speech Tagger for Swedish. Northern European Journal of Language Technology, Vol. 3, pp 1–18 (2013)