Ontology-Driven News Classification with Aethalides

Wouter Rijvordt, Frederik Hogenboom and Flavius Frasincar*

Econometric Institute, Erasmus School of Economics, Erasmus University Rotterdam, Rotterdam, the Netherlands
E-mail: wouter@rijvordt.net, fhogenboom@ese.eur.nl, frasincar@ese.eur.nl
**Corresponding Author*

Received 13 February 2019; Accepted 22 November 2019;
Publication 30 November 2019

Abstract

The ever-increasing amount of Web information offered to news readers (e.g., news analysts) stimulates the need for news selection, so that informed decisions can be made with up-to-date knowledge. Hermes is an ontology-based framework for building news personalization services. It uses an ontology crafted from available news sources, allowing users to select and filter interesting concepts from a domain ontology. The Aethalides framework enhances the Hermes framework by enabling news classification through lexicographic and semantic properties. For this, Aethalides applies word sense disambiguation and ontology learning methods to news items. When tested on a set of news items on finance and politics, the Aethalides implementation yields a precision and recall of 74.4% and 49.4%, respectively, yielding an $F_{0.5}$-measure of 67.6% when valuing precision more than recall.

Keywords: News personalization, word sense disambiguation, ontology learning, semantic web.

Journal of Web Engineering, Vol. 18_7, 627–654.
doi: 10.13052/jwe1540-9589.1873

1 Introduction

The Internet is comprised of an ever-growing amount of information that is structured in such a way that it is easily usable and understandable for humans, but not for machines. The Semantic Web is designed to overcome this deficiency. A collection of languages, such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL), provide a way to enrich data with meta-data. Specialized software can use this meta-data to validate, present, and extend information.

Continuously updated news streams are an upcoming and increasingly popular source of information on the Web, and are published on websites and RSS feeds. This information is only lightly categorized: some sites contain news on a specific subject, while other news hubs categorize the news in general topics like "business" or "politics". This news is a relatively unstructured source of information, but could serve as a valuable input in financial applications [15, 16].

News analysts need to process as much relevant news as possible to be optimally informed, while minimizing the amount of irrelevant news items read. This can be achieved by a news processing or personalization system that selects relevant messages and filters out all irrelevant messages from news streams. Although processing speed is of paramount importance (e.g., in high frequency trading), high accuracy is equally important as it minimizes the risks associated with using inferred knowledge in financial applications.

A typical baseline approach would be a keyword-based system. However, major linguistic pitfalls cannot be tackled adequately. A semantics-based system for news analysis is hence required for accurate news personalization. Such a system should extract ontological components (i.e., concepts, instances, relations, and attributes) from a news message, exploiting the ontology's internal structure to generate the user's set of interesting concepts. The connections between these concepts and other news can then used to generate a list of relevant news items, which is a subset of all available news items. A news filtering system can use this process to automatically select interesting news items from a virtually endless stream of news.

In this article, we aim to determine how linguistic and semantic technologies can be applied in a financial decision support system that uses news items as its primary input. More specifically, we focus on the use of Word Sense Disambiguation (WSD) and ontology learning in a news filtering and personalization system that supports the decisions of financial professionals. The vast majority of the required (pre)processing steps for WSD, such as lemmatization, part-of-speech (POS) tagging, word frequency counting, etc.,

have been heavily researched, resulting in many high-performing off-the-shelf methods and tools, especially for English texts. Therefore, our research is primarily targeting the creation and use of ontology learning methods that extract knowledge to improve (news) filtering processes. Our main contribution is the development of a news filtering and personalization system that is an extension to the Hermes framework [14, 17, 20], i.e., Aethalides, which is fast, accurate, extensible, easily integrable, and generally well performing on the financial domain.

The remainder of this paper is organized as follows. Section 2 discusses the related work, followed by Sections 3 and 4, which elaborate on the Aethalides framework and its implementation, after which we evaluate our method in Section 5. Section 6 gives our conclusions and identifies future work.

2 Related Work

There is a great body of literature on the topics touched upon in this article. Because our work mainly focuses on the use of ontology learning and word sense disambiguation in the context of news personalization, in our endeavours, we review state-of-the-art approaches, methods, algorithms and systems related to these research areas.

2.1 News Personalization

Currently, popular news filtering systems, such as the multilingual EMM NewsExplorer [9] and the successors of the discontinued customizable Yahoo! Pipes as introduced by Yahoo [39], such as FeedsAPI [12] and dlvr.it [10] utilize simple filtering techniques, like keyword co-occurrence and matching. However, they fail to discover and exploit the semantic information contained in the news items.

SeAN [1] is a multi-agent adaptive system that allows for news selection and presentation by matching news items to topics that are hierarchically organized. Users are modeled through an estimation of their interests, domain expertise, receptivity, and life style. These estimates are then used to compute match scores (similarities) of users with predefined stereotypes. The topics associated to these stereotypes are used to select relevant news items for the user. SeAN monitors the activity of its users and utilizes this information to update the match between users and stereotypes and to revise the stereotypes themselves. Additionally, the system has the ability to target its users with personalized advertisements.

SemNews [21] aims to provide a structured representation of the meaning of news. It aggregates newspublished in various places on the Internet. The focus of the project is on supplying live and updated information. After the preprocessing stage, which involves part-of-speech tagging, morphological analysis, and recognition of names, dates, acronyms, and named entities, SemNews uses an ontology-driven environment to perform syntactic and semantic analysis. The results are then translated into OWL. Last, the discovered knowledge is expanded by analyzing the geographical and temporal properties of the produced RDF triples and generating additional ones by applying reasoners. SemNews includes various visualizations and editors for its knowledge base.

PlanetOnto [11] is a passive news collector that uses news items to build a knowledge base that supports text annotation and customizable alerting. MyPlanet [24], an extension of PlanetOnto, is equipped with advanced search heuristics and user profiling. However, the classification process of the news items is not automated and the system does not support visualization of its knowledge base, which impedes the users to have a good understanding of the available data.

The Knowledge and Information Management platform (KIM) [34] provides services and an infrastructure for generating, indexing, and retrieving semantically annotated documents. KIM relies on the KIM Ontology (KIMO), a minimal, yet sufficient, ontology suitable for open-domain general-purpose semantic annotations. KIM's information extraction focuses on named entity recognition and does not contain components dedicated to the disambiguation of entities.

The systems presented in [33] and [32] show resemblances with our work. These semantic services aggregate news articles and blogs, respectively, using Semantic Web technologies. For this, ontologies are populated with news data, but – in contrast to our approach – the frameworks are based on generic ontologies rather than a domain ontology. Therefore, our approach is able to index and query at a finer level of granularity. On the other hand, by reusing existing ontologies, the works of [33] and [32] are able to publish directly into the Linked Open Data cloud.

2.2 Word Sense Disambiguation

Any natural language features words that have multiple meanings, which can be determined using their contexts. Even when the part-of-speech of a word is known, the word sense can often still be manyfold. In the field of Artificial

Intelligence, the process of assigning meanings to words, i.e., Word Sense Disambiguation (WSD), has been actively researched since the sixties of last century. We will refrain from a highly detailed overview of WSD approaches, as many articles, such as the work of Ide and Véronis [19], already carefully survey disambiguation approaches and their various aspects.

Generally, WSD techniques can be classified as (un)supervised corpus-based or knowledge-based methods. Corpus-based methods, such as Naïve Bayes, k-Nearest-Neighbour, Adaboost, and Support Vector Machines, are the result of applying machine learning theory to WSD, and are fully based on the existing corpus. Knowledge-based methods, on the other hand, use some form of external knowledge, such as semantic lexicons [13] to determine word senses or even sentiment [18]. The advantage over corpus-based methods is that the latter are only applicable to those words and senses that occur in the available annotated corpora, while knowledge-based methods can be used on any unrestricted text. Techniques in this category include handcrafted rules, methods that use dictionary definitions, inspired by Lesk [26] and methods that use similarity measures, and language-based heuristics, like Most Frequent Sense and One Sense Per Discourse.

Current state-of-the-art methods for WSD include, but are not limited to *GAMBL* (Genetic Algorithm for Memory-Based Learning) [8], a supervised corpus-based method that uses a genetic algorithm for parameter optimization, *SenseLearner* [29], which is a supervised corpus-based method that constructs collocation and contextual models for predefined word categories, and *SSI* (Structural Semantic Interconnections) [31], a knowledge-based approach to WSD that disambiguates words by evaluating word connectivity after generating a labeled directed graph representing the context of each sense of a word.

2.3 Ontology Learning

The advent of the Semantic Web has fueled the development of the field of ontology learning [3]. The corner stones of the Semantic Web are ontologies, which are formal specifications of knowledge that represent a set of concepts together with instances, relations, and attributes of these concepts. Ontologies are used in information architecture, library sciences, and software engineering, and can be created from structured data such as dictionaries, knowledge bases, and schemata, but can also be learned from unstructured text [6, 28], like news items.

Ontology learning methods can be classified based on the nature of their input. They distinguish between methods using text, dictionaries, knowledge

bases, semi-structured schemata, and relational schemata. In light of the rise of the Semantic Web and the transition from the World Wide Web, ontology learning from (unstructured) text has become increasingly popular.

The task of ontology learning can be split into a layer cake of tightly related, sequential subtasks [4]. Most of the existing ontology learning systems for textual inputs focus on concept formation and concept hierarchies, while some are able to extract uncommon relations and can detect synonyms. Naturally, term extraction in text-based ontology learning systems is often accomplished by parsing and syntactic analysis. Extraction of relational hierarchies, axiom schemata, and general axioms is seldomly explored.

There is an ever-growing amount of ontology learning systems that use natural language texts as main inputs. Examples of well-known systems are *Text-To-Onto* [27] for the extraction of non-taxonomic relations, the *On-toLT* [5] plugin for the widely used ontology-editor Protégé [36], the *Mo'K Workbench* [2] platform for the creation, comparison, evaluation, and elaboration of clustering methods that create conceptual hierarchies, and *Isolde* [38] for creating and populating ontologies based on Web resources.

3 Framework

This section introduces the Hermes and Aethalides frameworks, which constitute the semantically enabled news filtering and personalization system.

3.1 The Hermes Framework

The Hermes framework, as introduced by Frasincar et al. [14], comprises a sequence of steps required for serving personalized news. As depicted in Figure 1, four main procedures can be distinguished:

1. *News Classification*: The first processing step associates news items to concepts from Hermes' domain-specific financial ontology (detailed in Table 1), which is created and maintained by domain experts. The concepts in this domain ontology are linked to synsets from a semantic lexicon. The classification process determines which of the words in a news item are lexical representations of synsets in the domain ontology. A prerequisite for optimal performance is that such a lexicon has a good coverage of the target domain. However, the semantic lexicon should also include general synsets and synsets associated to domains other than the target domain in order to enable accurate classification, because relevant and irrelevant senses need to be distinguishable. For example,

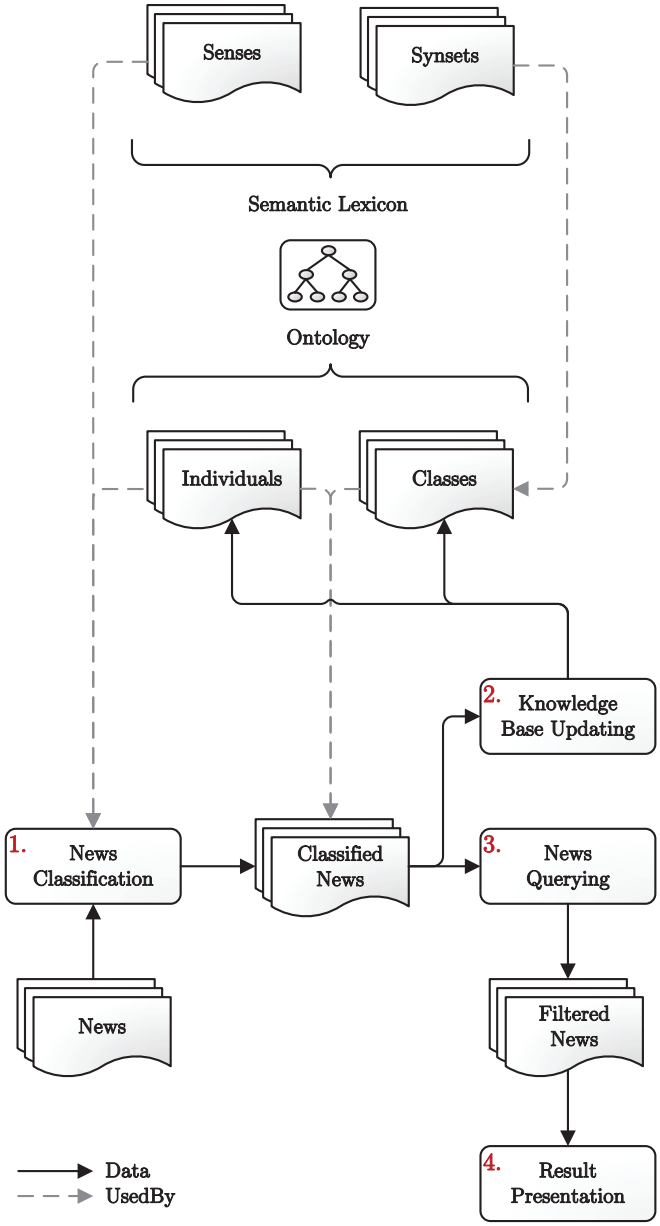

Figure 1 Hermes framework architecture.

Table 1 Details of the Hermes financial ontology

Description	Value
Number of classes:	69
Number of object properties:	22
Number of datatype properties:	4
Number of individuals:	416
DL expressivity:	$\mathcal{ALUIN}(\mathcal{D})$

within the financial domain, news messages discussing the company "Apple" should be distinguishable from the ones only discussing the fruit. Note that the domain ontology is not necessarily a strict subset of the semantic lexicon, because the granularity of the semantic lexicon and the domain ontology might not be the same. Therefore, concepts that have multiple or no synsets attached might exist. If a concept has no synsets it is not used in the News Classification step. However, it can be used in subsequent steps (e.g., Knowledge Base Updating).

2. *Knowledge Base Updating*: News items may contain knowledge that is relevant to the domain of the ontology and therefore the user, but that is not yet included in the ontology. In the knowledge base updating process, new facts are retrieved from the classified news items and incorporated into the domain ontology, so that it reflects all that is currently known. Each piece of knowledge that triggers such a modification is called an event. Event extraction rules are applied to detect such events in newly classified news items.

   These event rules are based on lexico-semantic patterns, that are manually or automatically constructed from concepts in the ontology and sections of text [20]. When text segments from news items match these patterns, the automatically recognized events are submitted to the user for validation. In order to prevent invalid or otherwise incorrectly recognized events from corrupting the ontology, the event updates are propagated to the knowledge base when the user explicitly marks them as valid. This change of the ontology is performed by rule actions [35], which are associated to event rules. If an event rule triggers and is considered to be valid, the event actions are executed. There are two types of actions: add and remove actions, which insert and delete information to the knowledge base, respectively. Note that updating existing knowledge is possible by associating a remove action, that deletes the old information, and an add action, that inserts new information, to an event rule.

3. *News Querying*: In the News Querying step, the user-formulated queries are executed to retrieve news items that interest the user. These queries are composed by allowing the user to select concepts from the domain ontology, and can include constraints on the concepts or their relations, as well as restrictions on news item time stamps. An interesting feature of the Hermes framework is that it takes into account concepts which are not directly selected by the user when suggesting news items. In fact, some of the suggested items may only have an indirect relation to the selected concepts, but are selected solely based on their associated concepts relations to the directly selected concepts.
4. *Results Presentation*: The final processing step handles the presentation of the results of the news querying process. These results are a selection of the available news items, which are firstly sorted by relevance degree, i.e., the number of relations found between the query concepts and the item, and secondly, by the item time stamps. The lexical representations of the query concepts are highlighted in the text of the item, offering a visual explanation to the user as to why the item was marked as relevant.

3.2 The Aethalides Framework

Aethalides is an extension to the Hermes framework that enhances the news classification process, i.e., the first step in the Hermes framework. Apart from performing a series of inevitable preprocessing procedures, Aethalides primarily disambiguates words senses to accurately classify the news items.

3.2.1 WSD preparation

Disambiguation cannot be performed on the raw text of the news items, which in essence is a large sequence of characters. Therefore, the text must first be analyzed in order to make it suitable for WSD. This includes determining which characters form words and sentences, among others. Aethalides uses a WSD algorithm, discussed below, which requires certain properties, like parts-of-speech and lemmas, to be known beforehand. The processes that prepare the text for WSD are depicted in Figure 2, and perform the following tasks:

1. *Tokenizer*: Splits the corpus in character groups representing words, numbers, symbols, punctuation, and blanks.
2. *Named Entity Recognizer*: Identifies individuals, i.e., instances of concepts from the domain ontology.

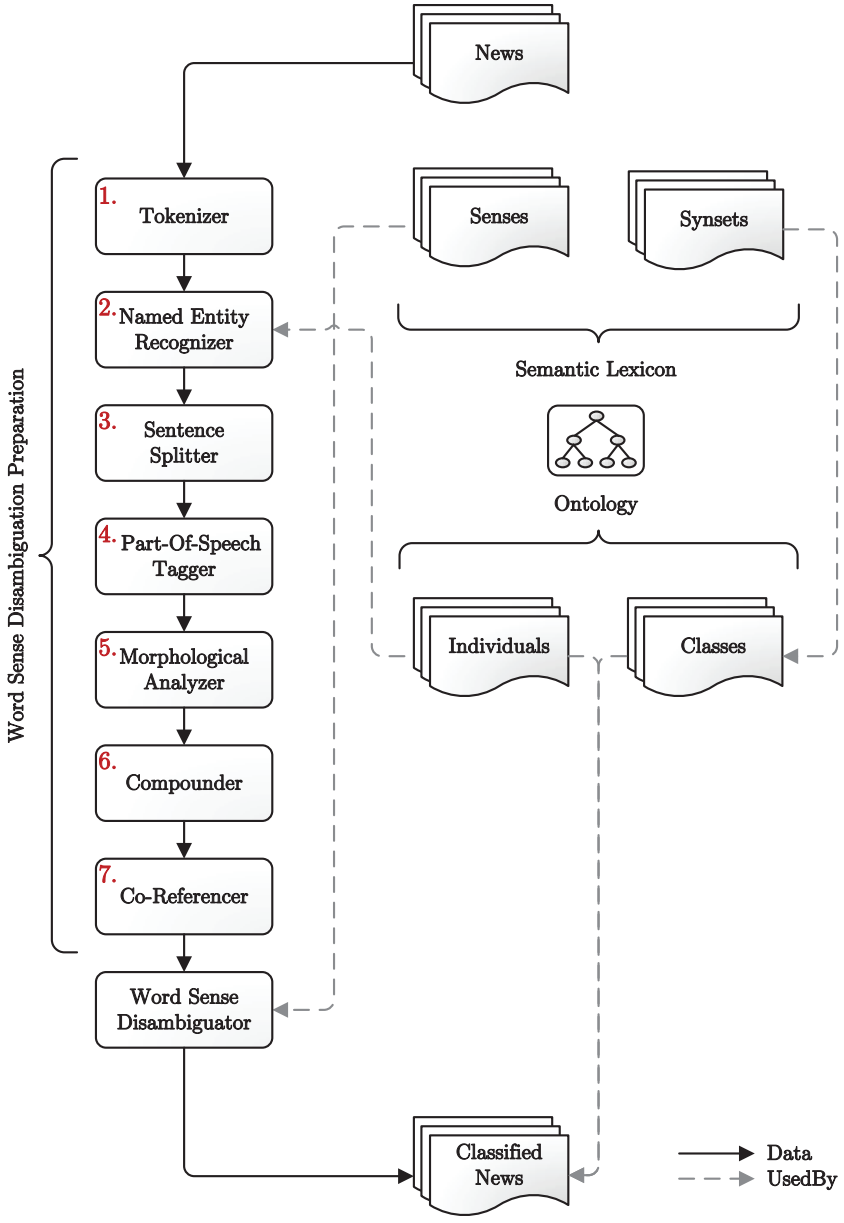

Figure 2 Aethalides framework architecture.

3. *Sentence Splitter*: Determines sentence boundaries by evaluating punctuation marks and capitalization.
4. *Part-of-Speech Tagger*: Determines the part-of-speech of each word using statistical models.
5. *Morphological Analyzer*: Identifies the lexeme of each word and reduces it to its lemma by removing conjugation and inflection.
6. *Compounder*: Recognizes compound words (lexemes that consist of multiple words) and their lemmas. It ensures that compound words are identified as such and are not processed as multiple single words, e.g., "combustion engine" should be recognized as a single concept and not as the two concepts "combustion" and "engine".
7. *Co-Referencer*: Identifies references to previously found entities while distinguishing between orthographic and pronominal co-references.

3.2.2 WSD algorithm

Aethalides employs the meta-algorithm Structural Semantic Interconnections (SSI) [31] for disambiguating word senses. To determine the most likely word sense, the SSI algorithm makes use of a semantic lexicon and the word context. The context of a word can be defined in multiple ways, e.g., the sentence, the paragraph or, if it is relatively small, the entire text in which the word occurs. Another option is a distance-based (window-based) approach, where the context consists of all n words surrounding a target word. SSI calculates the similarity between the senses of the words in the context and all possible senses of the ambiguous word. The sense that is most similar to its context is chosen. Words are disambiguated one-by-one, and the chosen senses are used as the context of ambiguous words that are processed in subsequent iterations. Therefore, SSI may yield different results if the words are processed in a different order. The algorithm depends on the presence of monosemic words in the context of the first few processed words to seed the disambiguation, because otherwise there are no senses in the context of those first words to compare a subsequent word's similarity with. If no monosemic words are present, ambiguous words are disambiguated at random or according to heuristics such as using the most common sense in general speech. This decreases the accuracy of the algorithm, yet it is a necessary step to acquire a seed for the algorithm.

Unlike other WSD methods like GAMBL and SenseLearner, SSI does not yield a black box kind of result, because the similarities can be inspected by an expert. This means that it is possible to trace why SSI chooses a particular sense for a word. Another advantage is that SSI does not have

to be trained, whereas other techniques need a training phase before they perform reasonably well. SSI can be applied without a costly initialization phase, making it a well suited algorithm for a time-constrained task. Last, SSI does not need to be retrained periodically. When a word is disambiguated, it can be used in further runs of the algorithm and may help disambiguating other words. Other WSD algorithms, like SenseLearner, require a complete refit of their models, when the corpus or the semantic lexicon changes.

3.2.3 Result processing

Once the senses of the words in a news item are determined, Aethalides establishes links between classes in the domain ontology and the news item based on news item word senses and existing links between classes in the domain ontology and synsets in the semantic lexicon. Additionally, domain ontology individuals are linked to named entities in the news item as found by the Named Entity Recognizer. Moreover, new individuals are added to ontology classes if needed.

4 Implementation

The implementation of the Aethalides framework introduced in Section 3.2, consists of four main components: the pipeline of processing steps, a semantic lexicon, the domain ontology, and (classified) news articles. Conform the framework specifications as illustrated in Figures 1 and 2, the ontology, semantic lexicon, and news articles are shared with the other processes in Hermes.

4.1 Pipeline

The GATE (General Architecture for Text Engineering) [7] framework is the defacto standard for natural language processing and information extraction tasks. It includes an integrated development environment that can be used to (manually) create, inspect, change, and compare annotations in various corpora. The system supports text files of various types, including HTML, XML, Doc, and PDF. Any meta-data contained in these documents is converted to GATE style annotations to allow further processing. For example, HTML anchor elements, normally used for hyperlinks to documents, are converted to an annotation containing the URL of the referenced document. GATE includes an extensible framework and API for natural language processing and information extraction, allowing it to be easily integrated and extended in other applications.

A GATE application consist of a series of processing resources that create and modify annotations attached to the input corpus. Each of these resources is applied sequentially to the corpus and has access to the output (annotations) of its predecessors. The default GATE distribution contains many processing resources, and additional custom components can be easily created. ANNIE (A Nearly New Information Extraction System), GATE's default information extraction system for unstructured English corpora, which also acts as a technology demonstrator for the GATE framework, is an example of an application built from processing resources. A powerful feature of GATE is JAPE (Java Annotation Patterns Engine), which provides finite state transduction over annotations based on regular expressions. The JAPE-language allows the creation of annotation modifying rules. Such a rule consists of two parts: the left hand side (LHS) and the right hand size (RHS). The LHS is a regular expression (written in a language suitable for the graph-based GATE annotations), which defines the prerequisites of firing the rule. The RHS is either a simple annotation assignment or a section of Java-code, which gives JAPE access to the full power of a programming language to modify annotations. JAPE also provides macros and priority control, which means multiple rules can share sections of their LHSs and RHSs.

The Aethalides pipeline is implemented as a GATE application. All its processing resources are provided by GATE, except for the *Compounder* and the *Word Sense Disambiguator*, which are custom resources.

4.2 Semantic Lexicon

Aethalides uses the JWNL [23] API for WordNet [13] semantic lexicon, which is commonly used for disambiguation tasks and natural language processing in general. WordNet 3.0 contains 155, 287 words mapped to 117, 659 synsets that are interconnected by semantic relations, thus creating a network of senses. WordNet contains only open-class words and is restricted to the English language. Other projects have created WordNet-like lexicons in other languages, but are out of scope in our current endeavours.

4.3 Compounder

The Compounder recognizes compound words, which are words that consist of multiple tokens as marked by the Tokenizer. These compounds must be carefully handled to make sure that their tokens are not processed as multiple single words. The effect of not handling compound words correctly can be substantial, as 64, 311 out of 155, 287 (41.43%) WordNet words are

compounds. Aethalides contains two different Compounders. The Pattern Compounder recognizes compounds in a text by looking for patterns in the parts-of-speech of the words. In order to determine these compound patterns and estimate the degree to which they occur in a semantic lexicon, we gathered all compound words from WordNet and used the Part-of-speech Tagger to determine the parts-of-speech of all their tokens. The part-of-speech patterns of the compounds were then tallied and sorted by occurrence, which yields 1, 478 patterns, of which the top 11 account for more than 1% of occurrences each. This means that a compound recognition pattern needs only to match those few part-of-speech combinations that occur frequently to get a high coverage.

The choice of a recognition pattern for the Pattern Compounder is a delicate one. If we use all discovered patterns, the accuracy of the Compounder would be very high, but we are at risk of overfitting with respect to the semantic lexicon used generate those patterns. Additionally, some of these patterns, like VB NN, are very likely to occur in a text without constituting a compound word. Therefore, we need a relatively simple (i.e., short) pattern that matches as many of the high scoring patterns as possible.

The second Compounder, the Brute Force Compounder, approaches the problem in a different way. As the name suggests, it applies brute force to find compounds starting with each word in the corpus. To this extent, each starting word is joined with the next word. Subsequently, the combination is validated using a compound lexicon. Subsequent words are iteratively added to the combination, until the maximum number of words, i.e., the number of words of the largest compound in the lexicon, has been reached. In the case of WordNet, the limit is set to 12. This strategy will cover many of the compounds in the corpus, but most of its checks yield negative results, and therefore a single check must be implemented as a cheap, optimized operation in terms of computing resources. Otherwise, the Brute Force Compounder is a very slow strategy. Since the Brute Force Compounder exhaustively checks all candidate compounds including those not covered by the Pattern Compounder, we expect it to perform better than the Pattern Compounder. The difference in performance depends on the relative occurrences in the corpus of those compounds not covered by the compounder.

4.4 WSD Life Cycle

In our implementation of the Aethalides word sense disambiguation procedures, we start out by loading the annotations that are the results of the

preparatory steps described in Section 3.2, i.e., sentence and token annotations, where tokens are defined as words and their parts-of-speech and proposed lemmas. Since the SSI algorithm only operates on one sentence at a time, the next step is to group the words by their sentences. Practically, this means a smaller memory footprint and a possibility for parallel processing. Then, the results of the morphological analyzer are double-checked against the semantic lexicon (in our case WordNet), as the morphological analyzer uses generalized rules to remove conjugation and inflection, but does not account for some words that have irregular conjugations and inflections.

The SSI algorithm requires that some words in a given sentence are unambiguous before disambiguating the others. In many sentences, monosemic words serve this propose, as they have only a single possible sense and are thus disambiguated by default. As noted earlier, some sentences, however, contain no monosemic words. In order to be able to apply SSI on these sentences, an alternative initialization strategy is required. In this strategy, the least ambiguous word in the sentence, i.e., the word with the lowest number of possible senses is forcibly disambiguated by choosing its first sense. The first sense has the highest probability of being the correct sense, because in WordNet, the senses are sorted by their relative frequency of occurrence.

During the application of the SSI algorithm to an arbitrary sentence, the possibility remains that the algorithm is unable to disambiguate some of the words. This occurs when the distances — which are computed using the Jiang and Conrath [22] similarity measure — between disambiguated and ambiguous words are larger than a predefined limit, thus preventing excessive processing times. In that case, we assume the similarity is negligible. In order to move the process along and not be forced to abandon the entire sentence, a fall-back strategy is required. We apply the same strategy as the alternative initialization strategy, i.e., the forced disambiguation of the least ambiguous word that is left.

Once the disambiguation procedure using the SSI algorithm has completed processing a sentence, we store the senses, and optionally their glosses and disambiguation causes (monosemic, alternative initialization, SSI, and fall-back) into the existing Token annotations of the corpus.

4.5 Ontology

The domain ontology in Hermes, which is used by Aethalides, is represented as an OWL graph and is accessed using the Jena [37] Semantic Web framework. Aethalides uses the same OWL graph to store the news articles

themselves. This makes it easy to create and maintain the links between the classified news articles and concepts in the domain ontology. It also improves inter-operability with other systems and removes the needs for a separate data store for the news articles.

The concepts in the domain ontology and the news articles are represented in the OWL graph as classes. The attributes of the articles, such as author, publisher, publication time/date, title, and the full text of the article are stored as properties. The relations between the domain concepts and the articles, which are created by the News Classification process, are not modeled as properties of the domain concepts or the articles, but as a separate Relation class. This class has two main properties and the objects of one of these properties are the domain concepts and the object of the other property are the news articles. This approach [14] allows the modeling of properties on properties, e.g., a property that denotes the strength of the relation between an article and a domain concept. The relations between the domain concepts and synsets from the semantic lexicon have no need for this feature and are therefore represented as properties of the concepts with the synsets as the allowed range. The synsets can be directly referenced in this way using the OWL representation of WordNet.

5 Evaluation

In order to demonstrate the effectiveness of the Aethalides component of the Hermes system, we evaluate its performance against real-world data. Because Aethalides consists of a sequence of processing steps, the first series of experiments evaluate each processing step separately and independently from the other steps. For this, we apply each processing step to an ideal version of its input data created from a golden standard, as opposed to using the actual output of the previous step. In this way, any deviation of the desired result of a given step must be attributed to the process itself, and not to the propagation of errors inherited from preceding steps. This approach pinpoints those steps that contribute the most to any loss in performance of the entire sequence, and additionally allows for the independent analysis of the intermediate results of each step. The second series of experiments assesses the overall performance of the processing steps by feeding the processing pipeline a prepared test set and comparing the outcome with a result set that consists of known good examples of Aethalides' target objects, i.e., links between news items and an ontology, that describe which ontology concepts have been found in the news items and their positioning.

The remainder of this section discusses the corpora used as data set for our various experiments, the employed evaluation measures, and the experimental results.

5.1 Data Set

A perfect disambiguation system will approach the performance of annotations created by humans. In order to measure performance accurately, an annotated corpus must consist of annotations that are independently identified by multiple humans based on their inter-annotator agreement. For our purposes, there is a variety of usable, annotated corpora readily available.

The Brown Corpus [25] is a general American English corpus that consists of just over a million words divided into 500 samples of about 2, 000 words each taken from texts first published in 1961. SemCor [30] is a subset of the Brown Corpus consisting of 352 samples. Each word (that had an applicable part-of-speech) in the samples is semantically tagged with WordNet synsets, making the corpus an effective test set for WSD applications. However, the selected data set for our experiment will only include the types of text that the system will likely encounter when applied to the target domain, i.e., political and financial news. SemCor is a well-known corpus for WSD tasks, which is why it is used in these experiments. It has a inter-annotator agreement of 78.6%. This might have been higher, if not for the large number of senses for words in WordNet.

5.2 Performance Measures

In the field of Information Retrieval, the two classic performance measures precision and recall are commonly used to determine the quality of an IR system on a given test set. Precision (P) is defined as the number of correctly identified items as a percentage of the number of items identified. This measure takes into account only those items that are identified, i.e., missed or ignored items do not influence precision, and measures the performance of that identification. High precision means that the identifications made by the system are correct. In general, precision measures the correctness of the actions taken by the system, unlike recall, which measures the number of correctly identified items as a percentage of the total number of known correct items. Recall (R) measures how many of the items that should have been identified actually were identified. High recall means the system covers the entire sample space and does not miss identifications.

There is a trade-off between precision and recall, since a system can easily achieve 100% precision by identifying nothing and thus making no incorrect identifications. However, this means the recall is 0%, because none of the required identifications were made. Similarly, a system can achieve 100% recall by identifying everything, including all required identifications, but putting no effort in making good quality identifications. In this case the precision will be low, because many identifications are present, but incorrect. The F-measure (F_β) balances this trade-off by taking the weighted harmonic mean of *P* and *R*, where β is a weight adjusting the importance of precision and recall. Common values are $\beta = \frac{1}{2}$ (precision is more important), $\beta = 2$ (recall is more important) and $\beta = 1$ (precision and recall are equally important).

Since an annotation to the corpus is uniquely identified by the combination of feature type, content, and span (beginning and ending character in the corpus) properties, the actual annotations created by the system and the expected annotations from the test set can compared to each other, resulting in correct, incorrect, partially correct, or missing outcomes. The performance measures can be further refined to suit the classes of results. The main difference is the impact of partially correct identifications. Strict precision, recall, and F-measures P^-, R^-, F^- treat these as incorrect, while lenient measures P^+, R^+, and F^+ treat them as correct. Average measures $P^\pm$, $R^\pm$, and $F^\pm$ consider partially correct identifications as half a correct identification.

5.3 Experimental Results

The consolidated results of the experiments per processing step, and the experimental results of the entire pipeline are displayed in Tables 2 and 3, which provide insights into the precision, recall, and F-measures.

5.3.1 Tokenizer

The Tokenizer performs very well with a strict F-measure touching the 90% mark. However, it does have some problems with punctuation. Most notably, the dot signifying an abbreviation is often incorrectly interpreted as a separate token as if it were a sentence terminator. Also, the abbreviation dot sometimes incorrectly breaks up a token, for example “C.I.A.” should be a single token and not three. Last, the Tokenizer is unable to recognize tokens featuring white spaces, thus stressing the need for a Compounder. Unfortunately, since the Tokenizer is the first processing step, this does adversely affect its score.

Table 2 Consolidated precision and recall in percentages

Component	P^+	$P^\pm$	P^-	R^+	$R^\pm$	R^-
Tokenizer	91.3	88.4	85.5	100.0	96.8	93.6
Named Entity Recognizer	60.2	54.1	47.9	61.1	54.8	48.6
Sentence Splitter	94.7	88.4	82.1	98.2	91.6	84.9
Part-of-speech Tagger	91.8	91.8	91.8	90.2	90.2	90.2
Morphological Analyzer	96.0	96.0	96.0	96.0	96.0	96.0
Co-Referencer	84.7	82.3	79.9	72.7	70.7	68.7
Pattern-Based Compounder	29.2	23.2	17.2	58.1	45.7	33.3
Brute Force Compounder	67.1	66.6	66.1	84.8	84.2	83.6
Word Sense Disambiguation	57.9	57.9	57.9	100.0	100.0	100.0
Full Pipeline	77.5	74.4	71.3	51.5	49.4	47.4

Table 3 Consolidated F-measures in percentages

Component	$F^+_{0.5}$	$F^\pm_{0.5}$	$F^-_{0.5}$	F^+_1	$F^\pm_1$	F^-_1	F^+_2	$F^\pm_2$	F^-_2
Tokenizer	93.0	90.0	87.0	95.5	92.4	89.4	98.1	95.0	91.8
Named Entity Recognizer	60.4	54.2	48.1	60.6	54.4	48.2	60.9	54.7	48.4
Sentence Splitter	95.3	89.0	82.6	96.3	89.9	83.4	97.4	90.9	84.3
Part-of-speech Tagger	91.5	91.5	91.5	91.0	91.0	91.0	90.5	90.5	90.5
Morphological Analyzer	96.0	96.0	96.0	96.0	96.0	96.0	96.0	96.0	96.0
Co-Referencer	82.0	79.7	77.4	78.2	76.1	73.9	74.8	72.7	70.6
Pattern-Based Compounder	32.4	25.7	19.0	38.7	30.6	22.6	48.3	38.1	27.9
Brute Force Compounder	70.0	69.5	69.0	74.9	74.4	73.9	80.6	80.0	79.4
Word Sense Disambiguation	63.2	63.2	63.2	57.9	57.9	57.9	87.3	87.2	87.2
Full Pipeline	70.4	67.6	64.8	61.9	59.4	56.9	55.2	53.0	50.8

5.3.2 Named entity recognizer

According to information extraction standards, the Named Entity Recognizer does not have a good performance, with all lenient measures residing in the lower 60's, and strict measures even touching regions below 50%. This means that this process cannot be automated, because full automation will yield many wrong links between text and ontology and many invalid individuals in the ontology, which would contaminate the ontology with incorrect data. Aethalides takes a semi-automatic approach, which means that the system will ask the user to confirm the validity of these links and individuals before creating them. Aethalides operates on news sources, which can contain a

great number of named entities, not all of which are relevant. Therefore, the system considers a named entity as relevant only if it is detected in three or more separate parts of the text.

5.3.3 Sentence splitter

The Sentence Splitter, like the Tokenizer, has some problems with abbreviations. The impact of this error on the Sentence Splitter is greater than on the Tokenizer, because dots are also common characters for signaling the end of a sentence. A single misinterpreted dot may yield multiple incorrect, partially correct, or missing annotations. Also, the Sentence Splitter has problems with quoted text embedded in a sentence (and thus having a grammatical role in that sentence). These types of errors yield partially correct results and have little impact on the WSD. The exact boundaries of the sentences are not critical to WSD, because of the implicit semantic connection of sentences that are near each other in the corpus. Overall, the performance of the Sentence Splitter is near-perfect, considering its lenient measures around 95%.

5.3.4 Part-of-speech tagger

The Part-Of-Speech Tagger is highly accurate with results over 90%. However, the experiment only covers those parts-of-speech required by subsequent processing steps, i.e., nouns, verbs, adverbs, and adjectives. Performance on other parts-of-speech are not tested, since they are not required. Note that partially correct parts-of-speech are not possible, because the Part-Of-Speech Tagger adds a single annotation to existing tokens, as created by the Tokenizer. This means that the output of the tagger for the purpose of the experiment is either correct, incorrect, or missing.

5.3.5 Morphological analyzer

The Morphological Analyzer has problems with phrasal verbs, which are verbs plus complementary postpositions. For example, in the sentence “He carried on, despite the warning”, the verb “carry” has the complementary postposition “on”. Note that the verb is inflected, but the complementary postposition is not. Together they form a phrasal verb, which semantically forms a single unit. The Compounder recognizes such a phrasal verb. However, since the Compounder needs the lemmas of those words as provided by the Morphological Analyzer to determine what to compound, it cannot help the Morphological Analyzer in removing conjugation and inflection. Despite this shortcoming, the Morphological Analyzer has a near perfect performance with all measures over 95%. Nearly all errors are caused by phrasal verbs.

Like the Part-Of-Speech Tagger, the Morphological Analyzer only adds an additional annotation to existing tokens, thus making partially correct results impossible.

5.3.6 Compounder

The high number of incorrect annotations created by the Pattern Compounder can be explained by the relatively wide compound pattern, which covers just over three quarters of all possible compounds. The Pattern Compounder therefore is expected to have a recall of about 75% of the recall of the Brute Force Compounder. In reality, however, at 45.7%, the Pattern Compounder performs worse than that. The precision of the Pattern Compounder is also much lower than the precision of the Brute Force Compounder. This means that the pattern used is too wide and captures too many false positives (incorrect results). Therefore, the Brute Force Compounder seems to be a better choice for Aethalides.

5.3.7 Co-Referencer

The Co-Referencer uses the syntactical and grammatical properties of the word in the corpus to find co-references. It is not semantically enabled and does not use any ontologies in its analysis, which negatively influences its performance. Also, there are some problems that have to do with writing style. Normally, when co-references occur in a text, they are relatively close together. However, some writers increase the distance between the (sets of) words that refer to the same entity so much that the Co-Referencer cannot detect this any more. The Co-Referencer scores $P^{\pm} = 82.3\%$ $R^{\pm} = 70.7\%$, which yields an F-measure of $F_1^{\pm} = 76.1\%$.

5.3.8 Word sense disambiguation

The result of the Word Sense Disambiguation experiment may seem anomalous, because of the zero missing and partially correct annotations. However, the missing score can be attributed to the fact the Word Sense Disambiguation has a built-in fall-back in case the regular disambiguation process yields no results. In this case, we assign the first sense in WordNet, i.e., the most common sense, meaning it will assign a sense to each word in its input data. Partially correct scores are also not possible, because a word sense is expressed as a natural number, which means that the word sense is either correct or incorrect.

When comparing Aethalides' WSD with other WSD systems, Aethalides performance is on par with most existing systems. For instance, the winner of the Senseval-2 competition for supervised WSD systems, achieved

a precision of 64%. Aethalides' score of 57.9% is well above Senseval's baseline (always choose the most frequent sense), which scored a precision of a mere 48%. Another common baseline to measure the added value of WSD systems is random sense picking, which scored a negligible precision of only 16%. Note that Senseval's baseline equals Aethalides' fall-back strategy. This means that Aethalides' efforts are rewarded with a 10% point accuracy increase over this baseline. Aethalides manages to achieve these results while retaining result traceability without the need for periodical retraining and without costly initialization procedures.

5.3.9 Full pipeline

Last, we evaluate the performance of all components as a whole, while using the Brute Force Compounder. The golden standard for this test consists of the information required to perform accurate classification of texts in the Hermes system, i.e., the location and nature of individuals (named entities) and classes (identified by lemma and word sense). This means that the golden standard is the combination of the golden standard of the Named Entity Recognizer and the Word Sense Disambiguator tests. The expected values of performance measures of the full pipeline test are therefore expected to be near the performance measures of those components, except for the recall of the Word Sense Disambiguator, since a recall of 100% is realistically unfeasible.

The precision of the full processing pipeline is higher than expected at $P^{\pm} = 74.4\%$. Recall, however, at $R^{\pm} = 49.4\%$, is lower than expected. It seems that errors do propagate through the pipeline causing lower performance. Since Aethalides is expected to be used on relatively long texts, like news articles, precision is more important than recall. Incorrect identification may pollute the ontology with incorrect data, while missing identifications may be compensated by other identifications elsewhere in the test. Therefore, $\beta = 0.5$ seems an appropriate value for computing the F-measure, yielding a final score of $F_1^{\pm} = 67.6\%$.

6 Conclusion

In this article, we have introduced the Aethalides extension of the Hermes framework, which is an ontology-based framework for building news personalization services. Aethalides adds improved classification of news items and limited ontology learning to Hermes. This is achieved by processing the news items through the Aethalides pipeline that is comprised of a sequence

of steps creating links between the news items and classes and individuals in Hermes' domain ontology. These links can be leveraged to select relevant news items for news readers (e.g., news analysts). Word sense disambiguation is performed using Structural Semantic Interconnections (SSI) algorithm based on the WordNet semantic lexicon and context sense similarities. We have evaluated the system on a set of news items selected from SemCor, resulting in an overall precision of 74.4%, a recall of 49.4%, yielding an $F_{0.5}$-measure of 67.6%.

We envision various directions for future research. First, the word sense disambiguation process, which is generally considered to be an open problem in information extraction and retrieval, can be improved using newly developed state-of-the-art disambiguation procedures. Moreover, the system would benefit from improved recognition of advanced relationships between concepts, e.g., time-limited relationships. Last, the main focus of Aethalides is on accurate classification of a large body of news items in order to present its users relevant new items, but the pipeline is dependent on its domain ontology. In future work, we would like to improve the ontology feedback loop in order to increase the descriptive quality of the domain ontology that is leveraged by Hermes and Aethalides, for example by automated concept detection or temporal reasoning.

References

[1] Liliana Ardissono, Luca Console, and Ilaria Torre. 'An Adaptive System for the Personalized Access to News'. *AI Communications*, 14(3):129–147, 2001.

[2] Gilles Bisson, Claire Nédellec, and Dolores Ca namero. 'Designing Clustering Methods for Ontology Building: The Mo'K Workbench'. In *Workshop on Ontology Learning at 14th European Conference on Artificial Intelligence (ECAI 2000)*, volume 31 of *CEUR Workshop Proceedings*, pages 13–19. CEUR-WS.org, 2000.

[3] Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. *Ontology Learning from Text: Methods, Evaluation and Applications*. IOS Press, 2005.

[4] Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. *OntoLT: Middleware for Ontology Extraction from Text*. IOS Press, 2005.

[5] Paul Buitelaar, Daniel Olejnik, and Michael Sintek. 'A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis'.

In *1st European Semantic Web Symposium (ESWS 2004)*, volume 3053 of *Lecture Notes in Computer Science*, pages 31–44. Springer, 2004.

[6] Philipp Cimiano, Andreas Hotho, and Steffen Staab. 'Learning Concept Hierarchies from Text Corpora Using Formal Concept Analysis'. *Journal of Artificial Intelligence Research*, 24(1):305–339, 2005.

[7] Hamish Cunningham. 'GATE, a General Architecture for Text Engineering'. *Computers and the Humanities*, 36(2):223–254, 2002.

[8] Bart Decadt, Véronique Hoste, Walter Daelemans, and Antal van den Bosch. 'GAMBL, Genetic Algorithm Optimization of Memory-Based WSD'. In *3rd International Work-shop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval–3) at 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, pages 108–112. Association for Computational Linguistics, 2004.

[9] DG-JRC and DG-Press. EMM News Explorer, 2018. From: http://emm.newsexplorer.eu/NewsExplorer/home/en/latest.html.

[10] dlvr.it. Smart Social Media Automation, 2018. From: https://dlvrit.com/.

[11] John Domingue and Enrico Motta. 'PlanetOnto: From News Publishing to Integrated Knowledge Management Support'. *IEEE Intelligent Systems*, 15(3):26–32, 2000.

[12] FeedsAPI. FeedsAPI: Create Full Text RSS Feeds Instantly, 2018. From: http://www.feedsapi.com/.

[13] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[14] Flavius Frasincar, Jethro Borsje, and Leonard Levering. 'A Semantic Web-Based Approach for Building Personalized News Services'. *International Journal of E-Business Research*, 5(3):35–53, 2009.

[15] Frederik Hogenboom, Michael de Winter, Flavius Frasincar, and Uzay Kaymak. 'A News Event-Driven Approach for the Historical Value at Risk Method'. *Expert Systems With Applications*, 42(10):4667–4675, 2015.

[16] Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, Franciska de Jong, and Emiel Caron. 'A Survey of Event Extraction Methods from Text for Decision Support Systems'. *Decision Support Systems*, 85:12–22, 2016.

[17] Frederik Hogenboom, Damir Vandic, Flavius Frasincar, Arnout Verheij, and Allard Kleijn. A Query Language and Ranking Algorithm for News Items in the Hermes News Processing Framework. *Science of Computer Programming*, 94, Part 1:32–52, 2014.

[18] Chihli Hung and Shiuan-Jeng Cheng. 'Word Sense Disambiguation Based Sentiment Lexicons for Sentiment Classification'. *Knowledge-Based Systems*, 110:224–232, 2016.

[19] Nancy Ide and Jean Véronis. 'Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art'. *Computational Linguistics*, 24(1):1–40, 1998.

[20] Wouter IJntema, Jordy Sangers, Frederik Hogenboom, and Flavius Frasincar. 'A Lexico-Semantic Pattern Language for Learning Ontology Instances from Text'. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 15(1):37–50, 2012.

[21] Akshay Java, Tim Finin, and Sergei Nirenburg. 'Text Understanding Agents and the Semantic Web'. In *39th Hawaii International Conference on Systems Science (HICSS 2006)*, volume 3, page 62b. IEEE Computer Society, 2006.

[22] Jay J. Jiang and David W. Conrath. 'Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy'. In *10th International Conference on Research in Computational Linguistics (ROCLING 1997)*, pages 19–33, 1997.

[23] JWNL. Java WordNet Library, 2018. From: https://sourceforge.net/projects/jwordnet/.

[24] Yannis Kalfoglou, John Domingue, Enrico Motta, Maria Vargas-Vera, and Simon Buckingham Shum. 'myPlanet: an Ontology-Driven Web-Based Personalized News Service'. In *Workshop on Ontologies and Information Sharing at 17th International Joint Conferences on Artificial Intelligence (IJCAI 2001)*, volume 47 of *CEUR Workshop Proceedings*, pages 44–52. CEUR-WS.org, 2001.

[25] Henry Kučera and Francis W. Nelson. *Computational Analysis of Present-Day American English*. University Press of New England, 1967.

[26] Michael Lesk. 'Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone'. In *5th Annual International Conference on Systems Documentation (SIGDOC 1986)*, pages 24–26. ACM, 1986.

[27] Alexander Maedche and Raphael Volz. 'The Ontology Extraction and Maintenance Framework Text-To-Onto'. In *Workshop on Integrating Data Mining and Knowledge Management (DM-KM 2001), at the 2001 IEEE International Conference on Data Mining (ICDM 2001)*, 2001. From: http://users.csc.calpoly.edu/~fkurfess/Events/DM-KM-01/Volz.pdf.

[28] Kevin Meijer, Flavius Frasincar, and Frederik Hogenboom. 'A Semantic Approach for Extracting Domain Taxonomies from Text'. *Decision Support Systems*, 62:78–93, 2014.

[29] Rada Mihalcea and Andras Csomai. 'SenseLearner: Word Sense Disambiguation for All Words in Unrestricted Text'. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 53–56. Association for Computational Linguistics, 2005.

[30] George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 'Using a Semantic Concordance for Sense Identification'. In *ARPA Human Language Technology Workshop (HLT 1994)*. Morgan Kaufmann, 1994.

[31] Roberto Navigli and Paola Velardi. 'Structural Semantic Interconnections: A Knowledge-Based Approach to Word Sense Disambiguation'. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1075–1086, 2005.

[32] Nikos Papadakis, Haridimos Kondylakis, Anastasios Kalaentzis, Ioannis Komporakis, Ioannis A. Deligiannis, Malvina Steiakaki, George Alexiou, and Xanthoula Atsalaki. 'BlogSearch: Semantic Services for Aggregating and Searching Blog Articles'. *International Journal of Semantic Computing*, 10(3):399–415, 2016.

[33] Koralia Papadokostaki, Stavros Charitakis, George Vavoulas, Stella Panou, Paraskevi Piperaki, Aris Papakonstantinou, Savvas Lemonakis, Anna Maridaki, Konstantinos Iatrou, Piotr Arent, Dawid Wiśniewski, Nikos Papadakis, and Haridimos Kondylakis. *Strategic Innovative Marketing*, chapter 'News Articles Platform: Semantic Tools and Services for Aggregating and Exploring News Articles', pages 511–519. Springer Proceedings in Business and Economics. Springer, 2017.

[34] Borislav Popov, Atanas Kiryakov, Damyan Ognyanoff, Dimitar Manov, and Angel Kirilov. 'KIM – A Semantic Platform for Information Extraction and Retrieval'. *Journal of Natural Language Engineering*, 10(3–4):375–392, 2004.

[35] Jordy Sangers, Frederik Hogenboom, and Flavius Frasincar. 'Event-Driven Ontology Updating'. In *13th International Conference on Web Information System Engineering (WISE 2012)*, volume 7651 of *Lecture Notes in Computer Science*, pages 44–57. Springer, 2012.

[36] Stanford Center for Biomedical Informatics Research. The Protégé Ontology Editor and Knowledge Acquisition System, 2018. From: http://protege.stanford.edu/.

[37] The Apache Software Foundation. Apache Jena – Version 3.1.1, 2018. From: http://jena.apache.org/.
[38] Nicolas Weber and Paul Buitelaar. 'Web-based Ontology Learning with ISOLDE'. In *Workshop on Web Content Mining with Human Language Technologies collocated with the 5th International Semantic Web Conference (ISWC 2006)*, 2006. From: http://www.dfki.de/dfkibib/publications/docs/ISWC06.WebContentMining.pdf.
[39] Yahoo! Inc. Pipes: Rewire the web, 2012. From: http://pipes.yahoo.com/pipes/.

Biographies

Wouter Rijvordt received his bachelor and master degrees in economics and informatics at the Erasmus University Rotterdam, the Netherlands, in 2013. His current research interests lie in the fields of machine learning and Big Data. Currently, he works at Eneco as a data engineer and developer.

Frederik Hogenboom obtained the master degree with honours in (computational) economics and informatics at the Erasmus University Rotterdam,

the Netherlands, in 2009. During his bachelor and master programmes, his published research mainly in the fields of the Semantic Web and learning agents. In 2014, he received the PhD degree in computer science from the Erasmus University Rotterdam, the Netherlands, where he focused on financial event extraction from news applied to algorithmic trading, disseminated in numerous publications. His current research interests and endeavours mainly go out to natural language processing and Semantic Web technologies.

Flavius Frasincar obtained the master degree in computer science from the Politehnica University Bucharest, Romania, in 1998. In 2000, he received the professional doctorate degree in software engineering from the Eindhoven University of Technology, the Netherlands. He got the PhD degree in computer science from the Eindhoven University of Technology, the Netherlands, in 2005. Since 2005, he is assistant professor in information systems at the Erasmus University Rotterdam, the Netherlands. He published numerous publications in the areas of databases, Web information systems, personalization, and the Semantic Web. He is a member of the editorial board of the Journal of Web Engineering, International Journal of Web Engineering and Technology, Decision Support Systems, and Computational Linguistics in the Netherlands.