A Marketplace for Web Scale Analytics and Text Annotation Services

Johannes Kirschnick[1], Torsten Kilias[1], Holmer Hemsen[1]
Alexander Löser[2], Peter Adolphs[3], Heiko Ehrig[3], Holger Düwiger[3]
(1) Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany
{firstname.lastname}@tu-berlin.de
(2) Beuth Hochschule für Technik Berlin, Luxemburger Straße 10, 13353 Berlin, Germany
aloeser@beuth-hochschule.de
(3) Neofonie GmbH, Robert-Koch-Platz 4, 10115 Berlin, Germany
{firstname.lastname}@neofonie.de

Abstract

We present MIA, a data marketplace which enables massive parallel processing of data from the Web. End users can combine both text mining and database operators in a structured query language called MIAQL. MIA offers many cost savings through sharing text data, annotations, built-in analytical functions and third party text mining applications. Our demonstration showcases MIAQL and its execution on the platform for the example of analyzing political campaigns.

1 Introduction

Data-driven services and data marketplaces are young phenomena (Schomm et al., 2013). Commonly known commercial examples are *Factual.com* for location-related data, *Amazon* as a marketplace for infrastructure and data, and Microsoft's *datamarket.azure.com*. These marketplaces offer common data flow and service characteristics (Muschalle et al., 2012). MIA is such a marketplace, it offers data driven services in the area of natural language processing in combination with business intelligence services for analyzing more than 600 million pages of the German language Web (from 2013-2014). The user can combine linguistic components with standard SQL operators in data execution pipes to create new data sets. The marketplace permits the user to keep the insightful data exclusive or to grant access permissions to other users for data, annotation services or pipelines. MIA enables to turn the Web, a particular rich source of information using extraction, processing and enrichment tasks (e.g., parsing or semantically annotating) into a valuable knowledge resources to support various business intelligence (BI) tasks. Such tasks are exemplified by the following questions from the area of political campaign analysis:

- What is the polarity (sentiment) associated with events found in online news media?
- Which newspapers are biased in the last years towards certain political parties?

The MIA project started in 2011, since then we observed many similar queries and demands from sales departments (monitor and identify leads that soon will buy a car), human resources (identify professionals with capabilities in text mining), market research (monitor the effectiveness of a campaign) as well as product development (incorporate feedback from customers into the development process).

1.1 Background

Transforming huge volumes of text into structured information that is suitable and useful for empowering applications in heterogeneous and previously unknown application areas is a challenge. MIA addresses this using own and third party research in the areas of text mining and distributed databases.

Single system for analytical and NLP tasks. The parse tree database (PTDB) by Tari (2012) stores dependency tagged sentences and retrieves subtrees with a key value index based on *Lucene*. MIA goes drastically beyond this functionality. It enables GATE (gate.ac.uk) and UIMA (uima.apache.org) developers to integrate their components. This greatly expands the available processing modules

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: System Demonstrations, pages 100–104, Dublin, Ireland, August 23-29 2014.

on the platform as existing pipelines can be easily incorporated into the marketplace. Furthermore, it gives users the ability to store and retrieve lexical, syntactic and semantic annotations. The MIA platform executes these components as user-defined functions in a massively parallel database, enables the aggregation of resulting annotations with other data sources, such as in-house relational databases, and conducts these operations, all in a single system. AnnoMarket (Tablan et al., 2013) is a marketplace for executing GATE pipelines and permits to upload third party extractors, but lacks the functionality to aggregate and join results. Rather, it requires the user to ship preliminary results to a database where the user conducts advanced analytical tasks. However, companies may eschew the high investment risk and the technical difficulties to ship and analyze hundreds of millions of documents.

Declarative languages for text mining. System-T (Chiticariu et al., 2010) includes AQL, a declarative language for defining extractors through rules. MIAQL extends this idea with SQL predicates for sub-queries, views, joins and for data crawling that operate on datasets represented as collections of nested tuples. MIA's data model incorporates a span-algebra and operators for processing sequences or trees (Kilias et al., 2013). MIA also leverages our research for extracting relations (Akbik et al., 2012) and for restricting attribute types (Kirschnick et al., 2013) which builds on the provided abstractions.

1.2 Our Contribution

MIAQL: Text mining and analytical tasks with a single structured query language. We demonstrate how our declarative language MIAQL empowers end users to describe text mining tasks. MIAQL also supports common SQL-92 operations, such as joins, views, sub-queries, aggregations and group by. We showcase how our parallel execution framework compiles a MIAQL query into a dataflow representation and finally into a set of map/reduce tasks (Dean and Ghemawat, 2008).

Data marketplace for annotation services. We showcase how data providers can integrate data sets, developers can share modules for text mining or data analytics, application vendors can offer specialized applications and analysts can use the raw or annotated data for ad-hoc queries.

Optimizations for Web-scale corpora. We showcase on 600 million pages from the German language Web how our parallel execution framework optimizes queries for distributed multi-core environments and for file-, key-value- and column-based datastores, such as HBase[1] or the Parquet.io[2] file format.

2 Demonstration

Scenario: How biased are newspapers? Consider an analyst with the hypothesis that certain newspapers bias their reports towards certain political parties. Listing 1 shows a set of queries for determining such a bias. For solving this task the analyst must first obtain a news corpus, either by crawling or by buying and downloading the documents. Next, the analyst formulates a query for spotting sentences that mention a politician and computes the subjectivity. Finally, the analyst counts these tuples over all documents, groups them by the newspaper/party, aggregates the subjectivity and sorts the results.

2.1 Execution Environment for the Obtain-Annotate-Join-Aggregate-Process

This and many other monitoring processes follow the same schema: Obtaining already exclusive data, such as fresh data, and make data even more exclusive by complementing (joining), aggregating and sorting it. Nearly every company uses similar processes in the data warehouse on relational data but not on text data from the Web. MIA supports this process now also on hundreds of millions of Web pages with the following technologies:

Simple SQL interface for analysts. MIAQL provides standard SQL-92 statements for data aggregation, such as COUNTS, MIN, MAX, AVG, GROUP BY, including sub-queries and views. As a result, MIAQL presents a familiar environment for data analysts and reduces the necessity to learn a new query language. For example, listing 1 shows aggregations for counting and averaging the subjectivity for each

[1] https://hbase.apache.org (Last visited: 16/5/2014)
[2] http://parquet.io (Last visited: 16/5/2014)

```
1  CREATE VIEW sentences AS
2  SELECT srcName,text,unnest(splitSentences(text)) AS sentence
3  FROM news2013;
4
5  CREATE VIEW sentencesWithPOSTags AS
6  SELECT *,annotatePOSTags(sentence) AS posTags
7  FROM sentences;
8
9  CREATE VIEW sentencesWithSubjectivity AS
10 SELECT *,isSentenceSubjective(text,sentence,posTags) AS
       sentenceSubjective
11 FROM sentencesWithPOSTags;
12
13 CREATE VIEW sentencesWithEntities AS
14 SELECT *,unnest(annotateEntites(sentence, posTags)) AS entity
15 FROM sentencesWithSubjectivity;
16
17 CREATE VIEW filterSentencesWithPoliticians AS
18 SELECT
       srcName,party,politician,booleanToInt(sentenceSubjective)
       AS subjectivity
19 FROM sentencesWithEntities AS s
       JOIN politicians AS p ON s.uri=p.uri;
20
21 SELECT srcName,party,politician,COUNT(*), AVG(subjectivity)
22 FROM filterSentencesWithPoliticians
23 GROUP BY srcName, party, politician;
```

Listing 1: The listing shows a set of views in MIAQL for correlating mentions of politicians in newspapers and detecting whether the reporting is subjective. The MIAQL optimizer compiles this into an execution plan, see Figure 1 for an example.

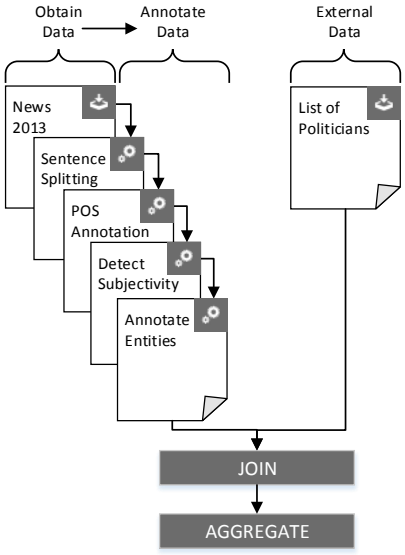

Figure 1: MIA supports user defined functions, optimizes equi, theta and cross joins into local, broadcast and repartitioning joins and supports common aggregate functions of SQL-92.

newspaper and politician/party pair. The query is transparently compiled into the data flow description language Pig Latin (Olston et al., 2008). This allows programs defined in MIAQL to be processed in a massively parallel way on a Hadoop system (Dean and Ghemawat, 2008). Figure 1 highlights the execution plan for the sample MIAQL query, which uses different data sources, processes them by executing a chain of annotators, joins the results and aggregates them to produce the final answer.

Reuse base annotators and user defined annotators. We observed from our users the need to enrich raw data with linguistic annotations at various levels. Users of the platform might be interested in named entities and their referents, topic labels or sentiment judgements for which they need to run specialized annotators. These annotators in turn might also depend on other levels of linguistic analysis. To prevent multiple processing of the same data with the same tools, the MIA platform crawls raw data and annotates it on ingestion at the most common linguistic levels. Multiple users can access these annotations across their projects and can thus save costs. For example, MIA performs **sentence splitting**, **tokenization**, **lemmatization**, **part-of-speech tagging**, **named-entity recognition** and a document-specific **topic detection** for each document. Other processing steps such as dependency parsing or entity linking (Kemmerer et al., 2014) can be applied on the fly at query time. Analysts may reuse this data in domain specific annotation functions. In the above listing the subjectivity analysis function reuses for example sentence boundaries and part-of-speech tags. The system executes such transformations as user-defined functions. Multiple functions are grouped into logical modules and registered on the platform for reuse, each with an associated description and a schema mapping from input to output tuples.

Complement Web data with in-house data. MIA permits users to upload in-house data to the distributed file system with restricted visibility. Our users join this data with the goal of complementing with existing data. For example, in the listing above a list of German politicians is joined with the list of extracted sentences to filter out all sentences that are not mentioning one of the defined politicians.

3 Lessons Learned

Data freshness, completeness and veracity. MIA's strongest asset is the ability to complement existing (possibly exclusive) in-house data with 'signals' from the Web, avoiding huge data set shipments.

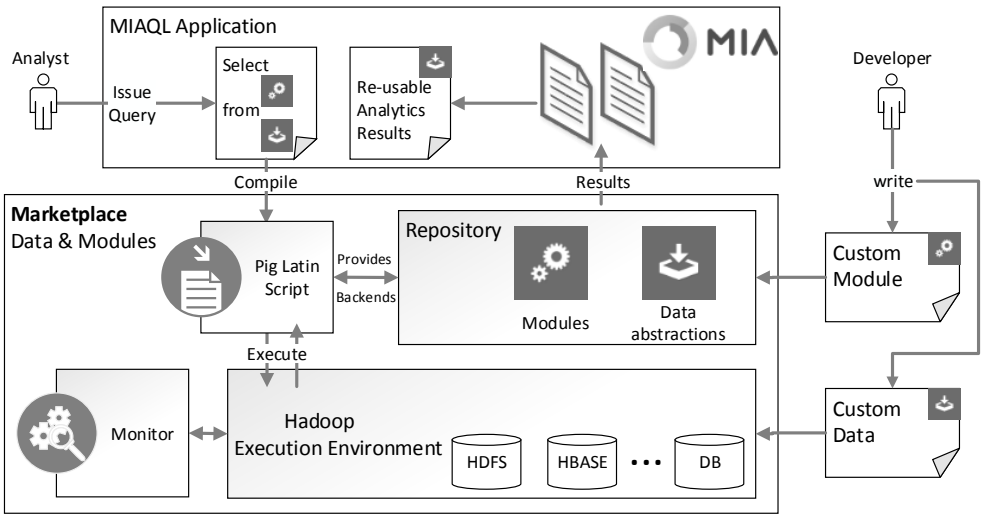

Figure 2: This figure shows the architecture of the marketplace. It contains the repository for data sets and text mining code modules. Developers can upload text mining modules, content providers can upload data sets and grant selected analysts reuse permissions. The application provides analysts with a Web interface for browsing the repository, authoring new MIAQL queries and downloading analytic results. Queries are transparently compiled into Pig Latin and executed on the distributed platform.

The Pig-Latin-based query processing engine provides users with answers within minutes. Currently, our users tolerate these answering times, for example for monitoring applications. However, users also often request fresh (minutes- hours) data, for tighter feedback loop integrations. Realizing short updates and low latency response times are research topics of the project, as well as delivering higher level insights beyond tabular reports, such as automated visualization and result clustering.

Cloud-based one-stop-shop marketplace vs. on-premise solution. Users can protect their data, functionality and processing results. However, some require running the MIA platform 'behind their firewalls'. Currently, they need to 'download' the processed results and execute the join in-house with their 'secret' data. For these users, we investigate operations for selecting the likely minimal and fresh but still highly complete data sets, such as Löser et al. (2013).

4 Current Status

The marketplace is currently available in private beta mode and accessible via a Web UI, which is displayed in Figure 3. Users can author, execute and monitor new jobs as well as inspect and download the results, while developers can create and upload new processing modules. New modules can be created by adhering to a simple tuple based processing interface defined by the platform, which greatly reduces the friction of writing new modules and leveraging existing text pipelines. The available public datasets are constantly expanded by continuously crawling the German Web, uploading new news articles and incorporating new social media posts.

While the project is targeted at providing large scale processing for German language texts, the core software platform is not tied to a particular language. All language specific contributions are packaged into separate data and processing modules. Supporting a new language sums up to creating and uploading new data and algorithm packages. More information as well as a demonstration video for MIA can be found at the project website `www.neofonie.de/forschung/mia`. We furthermore encourage requesting a demo login to the system.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. The work to build the demonstrator and the underlying technologies received funding from the German Federal Ministry of

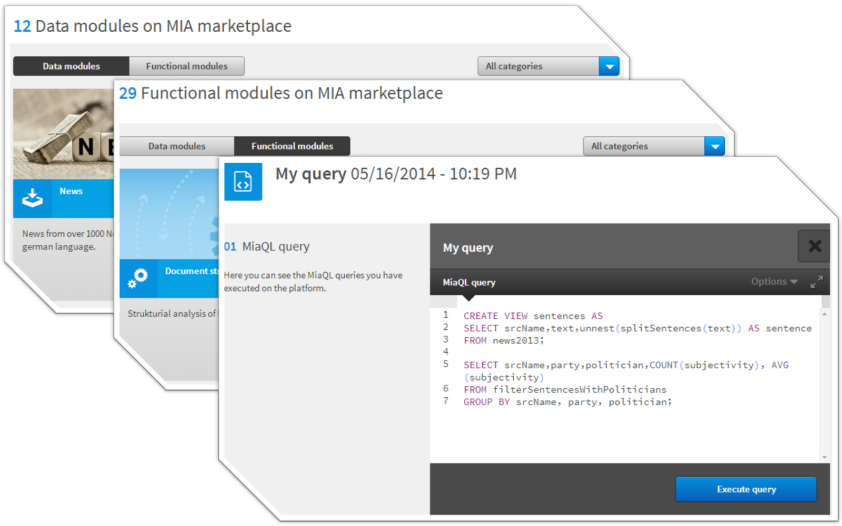

Figure 3: This figure shows the sophisticated execution environment presented by the MIA platform. The Web-driven UI supports the analysts with query authoring, data and function modules browsing as well as job monitoring and results download.

Economics and Energy (BMWi) under grant agreement 01MD11014A, “A cloud-based Marketplace for Information and Analytics on the German Web” (MIA).

References

Alan Akbik, Larysa Visengeriyeva, Priska Herger, Holmer Hemsen, and Alexander Löser. 2012. Unsupervised Discovery of Relations and Discriminative Extraction Patterns. In *COLING*, pages 17–32.

Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick Reiss, and Shivakumar Vaithyanathan. 2010. SystemT: An Algebraic Approach to Declarative Information Extraction. In *ACL*, pages 128–137.

Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM*, 51(1):107–113, January.

Steffen Kemmerer, Benjamin Großmann, Christina Müller, Peter Adolphs, and Heiko Ehrig. 2014. The neofonie nerd system at the erd challenge 2014. *SIGIR Forum*. To appear.

Torsten Kilias, Alexander Löser, and Periklis Andritsos. 2013. INDREX: in-database distributional relation extraction. In *DOLAP*, pages 93–100.

Johannes Kirschnick, Alan Akbik, Larysa Visengeriyeva, and Alexander Löser. 2013. Effective Selectional Restrictions for Unsupervised Relation Extraction. In *IJCNLP*. The Association for Computer Linguistics.

Alexander Löser, Christoph Nagel, Stephan Pieper, and Christoph Boden. 2013. Beyond search: Retrieving complete tuples from a text-database. *Information Systems Frontiers*, 15(3):311–329.

Alexander Muschalle, Florian Stahl, Alexander Löser, and Gottfried Vossen. 2012. Pricing Approaches for Data Markets. In *BIRTE*, pages 129–144.

Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins. 2008. Pig latin: a not-so-foreign language for data processing. In Jason Tsong-Li Wang, editor, *SIGMOD Conference*, pages 1099–1110. ACM.

Fabian Schomm, Florian Stahl, and Gottfried Vossen. 2013. Marketplaces for data: an initial survey. *SIGMOD Record*, 42(1):15–26.

Valentin Tablan, Kalina Bontcheva, Ian Roberts, Hamish Cunningham, and Marin Dimitrov. 2013. AnnoMarket: An Open Cloud Platform for NLP. In *ACL (Conference System Demonstrations)*, pages 19–24.

Luis Tari, Phan Huy Tu, Jörg Hakenberg, Yi Chen, Tran Cao Son, Graciela Gonzalez, and Chitta Baral. 2012. Incremental information extraction using relational databases. *IEEE Trans. Knowl. Data Eng.*, 24(1):86–99.