IJMIR manuscript No.
(will be inserted by the editor)

Information Extraction from Multimedia Web Documents: An Open-Source Platform and Testbed

David Dupplaw · Michael Matthews · Richard Johansson · Giulia Boato · Andrea Costanzo · Marco Fontani · Enrico Minack · Elena Demidova · Roi Blanco · Thomas Griffiths · Paul Lewis · Jonathon Hare · Alessandro Moschitti

Received: date / Accepted: date

Abstract The LivingKnowledge project aimed to enhance the current state of the art in search, retrieval and knowledge management on the web by advancing the use of sentiment and opinion analysis within multimedia applications. To achieve this aim, a diverse set of novel and complimentary analysis techniques have been integrated into a single but extensible software platform on which such applications can be built. The platform combines state-of-the-art techniques for extracting facts, opinions and sentiment from multimedia documents, and unlike earlier platforms, it exploits both visual and textual techniques to support multimedia information retrieval. Foreseeing the usefulness of this software in the wider community, the platform has been made generally available as an open-source project. This paper describes the platform design, gives an overview of the analysis algorithms integrated into the system and describes two applications that utilise the system for multimedia information retrieval.

Keywords multimedia retrieval · web analysis · text analysis · opinion analysis · image analysis · open-source software

1 Introduction

Most documents on the web are now inherently multimedia. Improving the retrieval of multimedia documents in terms of precision in response to more demanding queries requires improvements in knowledge extraction and indexing from multimedia data. This is a particularly challenging task as any articulations of knowledge are strongly influenced by the diversity of those articulating the knowledge. Expressed facts and opinions are sometimes influenced by biases that may correlate with the claimant's position on some axis of diversity such as the political spectrum or the spectrum of religious belief. Being aware of these biases is important for understanding the context in which knowledge is presented. As an example, in the presentation of news, some newspapers and broadcasters take a particular political or religious position and this can influence not only the way news is presented in text but also the way visual material is used in support of the news item. There have been high profile cases in recent years of image manipulation being used to support a particular view in a news story. The LivingKnowledge Project aimed to provide support for advanced multimedia retrieval tasks by developing a diverse, complimentary set of novel and state-of-the-art media analysis techniques (to, for example, detect and identify entities both in text and visual documents). Many of these modules are the result of research work in the project and this paper includes brief descriptions of key components. The mul-

David Dupplaw, Paul Lewis and Jonathon Hare
University of Southampton, United Kingdom, E-mail: {dpd | phl | jsh2}@ecs.soton.ac.uk

Michael Matthews and Roi Blanco
Barcelona Media, Spain E-mail: wanderingmike@hotmail.com | roi@yahoo-inc.com

Richard Johansson, Giulia Boato and Alessandro Moschitti
University of Trento, Italy E-mail: {johansson | boato | moschitti}@disi.unitn.it

Andrea Constanzo and Marco Fontani
CNIT, Italy E-mail: {andreacos82 | marco.fontani}@gmail.com

Enrico Minack and Elena Demidova
Leibniz Universität Hannover, Germany E-mail: {minack | demidova}@l3s.de

Thomas Griffiths
SORA, Vienna, Austria E-mail: tg@sora.at

timedia analysis modules are combined with a search engine and API to facilitate a versatile and extensible software platform, the Diversity Engine[1], for application development and multimedia information retrieval. The platform has been released as open-source software and can be downloaded from SourceForge.

The system is designed such that any self-contained, executable media analyser can be integrated as long as its output conforms to a fairly non-prescriptive information interchange format. The system provides the means for running the analysers over a large-scale Hadoop-based cluster providing horizontal scalability. It also provides an evaluation framework for both accuracy evaluation and regression testing of modules. On top of the analysis stage, the system uses Apache Solr[2] to provide search and retrieval capabilities over the extracted information and on which more substantial applications can be built.

This paper introduces the new platform and describes the main features of the architecture. It only summarises the available analysers for text, images and other multimedia data, giving pointers to the more detailed papers on these components where evaluations may be found. The paper concludes by showing how the platform is being used in two applications: The Time Explorer which extends the functionality of a web search engine and the Media Content Analyser where the system is applied to the analysis and retrieval of multimedia documents in support of social scientists identifying political trends and opinions in news media.

2 Diversity Engine Platform

Very few complete systems exist that provide a platform for text and multimedia analysis, indexing and retrieval. The GATE system [15] provides a pluggable architecture with a large number of text analysis filters and the Apache UIMA system is a complex framework designed for analysis tasks with limited APIs. We designed the DiversityEngine platform to process large numbers of web documents using analysers implemented using a diverse range of techniques and programming languages. An important aspect of the system is that it is open and easy to extend. The system was built in a collaborative research environment and it was important that it provided fast prototyping and testing using a wide array of implementation styles. Analysers are implemented as self-contained modules and provide multimedia (text, visual or overall document) analysis. The modules may output annotations at varying levels of complexity ranging from low-level feature-based information (such as parts-of-speech tags for text or visual features for images) or higher-level (more semantic) information such as opinion polarity for text or face detections for images.

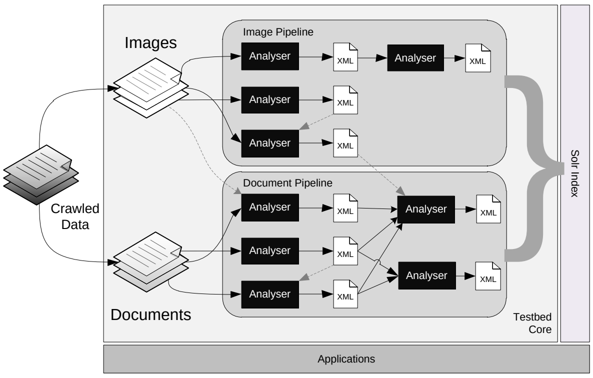


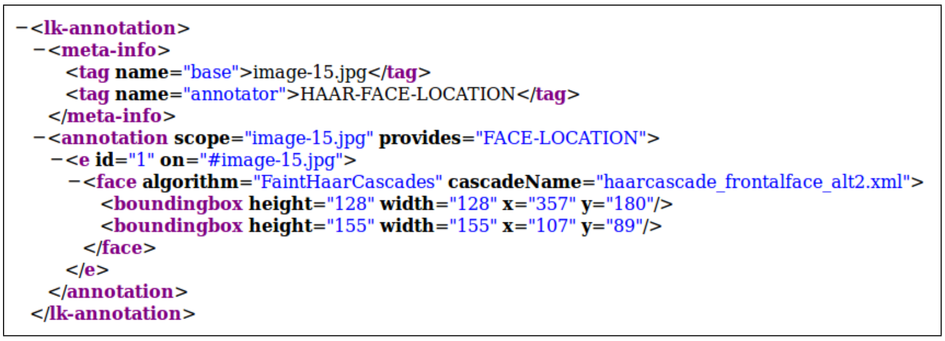
Fig. 1: A conceptual diagram of the DiversityEngine architecture. Documents are passed through image and document pipelines which perform annotation and write results to XML files. The XML files can be used directly by applications or converted into a Solr index which applications may utilise.

The core platform of the DiversityEngine is implemented in Java so it is platform independent. It provides the marshalling of data to and from the analyser modules as well as providing other useful methods that are made available to analyser modules that are implemented in Java. An overview of the architecture is seen in Figure 1.

The web-data to be analysed is provided to the system as WARC (Web Archive) files and the DiversityEngine core also provides extraction of the data from these files. The order in which analysers are executed on the data is determined by a simple configuration file provided by the user. Annotators are executed independently and serially as determined by the platform's configuration. When executing an annotator, the platform provides the annotator with the name of a directory in which it will find an XML representation of each web object and important images from the web document. These directories may also contain the output files from other modules which have already been executed. If the module utilises the Java APIs provided by the platform, the parsing of the XML files is automatically handled. The

[1] http://diversityengine.org

[2] http://solr.apache.org/



```
−<lk-annotation>
  −<meta-info>
      <tag name="base">image-15.jpg</tag>
      <tag name="annotator">HAAR-FACE-LOCATION</tag>
    </meta-info>
  −<annotation scope="image-15.jpg" provides="FACE-LOCATION">
    −<e id="1" on="#image-15.jpg">
      −<face algorithm="FaintHaarCascades" cascadeName="haarcascade_frontalface_alt2.xml">
          <boundingbox height="128" width="128" x="357" y="180"/>
          <boundingbox height="155" width="155" x="107" y="89"/>
        </face>
      </e>
    </annotation>
  </lk-annotation>
```

Fig. 2: Example of the XML wrapper for interchange where a specific image has been annotated with the bounding boxes of faces in the image.

modules write their output files into the same directory using a simple XML format which is mostly untyped to maximise flexibility [24]. An example of this XML format is shown in Figure 2. The software defines two discrete annotation pipelines: a document annotation pipeline and an image annotation pipeline. Each pipeline is configured by a configuration file which defines a set of annotation modules that will be run serially. The annotators in the image pipeline will receive different inputs to the annotators executed as part of the document pipeline. Having a distinct pipeline for image analysis eases the implementation of the image analysers by removing the need for the extraction of images from the document structure, as this is handled by the platform. Images can, of course, be analysed in the main document pipeline where the context in which they are presented will also be available.

Each annotation may contain individual elements, or entities, which are also referenceable through standard XML identifiers. The format also allows for references to be made to these elements in other annotation files to allow for linking. The explicit links between annotation files provides a dependency hierarchy of annotations. For example, text documents can be tokenised into words, the words can be assigned with word forms such as nouns and verbs, and the words can be aggregated into sentences. The hierarchy is implied: higher-level annotations (sentences) are based on lower-level annotations (words) . This annotation hierarchy is always rooted in either the initial text extraction that the platform extracts from the web document or the original document source (e.g. HTML). Extracting text from the original document is carried out by the core so that all analysers that require the text will receive the same text input and there will be no disparity between different extraction routines. For HTML documents, the text is extracted using a method of heuristic voting to determine the most likely element that may contain the main text for the article. Once extracted, this text is stored in an annotation file to which most of the text analysis annotations will refer. Annotations may reference specific parts of this initial text annotation by using character offsets from the beginning of the initial extracted text. Character offsets are not transferable back to the original source, so we have also created an analysis chain which allows text annotation offsets to be taken back into the original HTML file, which allows in-place manipulation of the original source. We use this for injecting RDFa content into the original article.

Similar hierarchies are built from image annotators where the image annotations are rooted in the actual image file. The annotation format does not specifically provide the means to refer to parts of an image (mainly because there are many ways to achieve this and we try to be non-prescriptive) but the annotation format is flexible enough to allow this to be included.

The looseness of the XML format has both advantages and disadvantages. Clearly, the flexibility means that the language, of any particular annotation, which may end up quite verbose, may be incomprehensible to some modules. It may also allow the divergence of two languages to describe the same ideas. However, the flexibility means that there is scope for any annotations to be represented, so there are no maintenance or longevity issues with the system. Also, the simplicity of the wrapper means that it is very easy to parse and write annotations for developers who are implementing their annotators in languages without strong XML libraries.

This hierarchy of annotations is somewhat implicit and does not necessarily provide the whole picture of what resources were used to create the annotation. So, the DiversityEngine provides an implementation of the Open Provenance Model [35] for annotating the annotations with provenance information thereby giving a concrete and explicit representation of the hierarchy. This information is automatically created for the Java analysers as they will use the core APIs to access annotations and resources. The graph for each annotation contains links that specify which other annotations (from other modules) or which resources (images or text) were used to create it. The provenance graph is appended to any annotations that are written by the annotator and can be recalled later for perusal or analysis. A complete provenance graph can be generated for an annotation by merging the individual graphs of the ancestors in the annotation tree.

2.1 Large-scale Analysis

While the DiversityEngine can run on a single machine for small collections, this is not desirable for large collections due to the time it takes to process such information. In general, the processing bottlenecks are the document and image analysis pipelines where some of the annotators can take up to 1 second per document[3]. However, because the processing of documents is independent, it is straightforward to parallelise the processing pipelines. We have successfully used the DiversityEngine on Hadoop[4], a popular open source mapreduce framework and the support for doing this is built directly into the platform.

The DiversityEngine includes several tools to enable and assist running jobs on Hadoop. As the DiversityEngine API presents directories of documents to processors rather than individual documents, any start-up time required by an analysis tool is incurred only once per directory. However, when processing in a cluster, the goal is to spread the load as evenly as possible between machines which is easier when jobs are smaller, so the choice of how to split the collection into subcollections for processing becomes important. When using WARC files for collections, this can be accomplished by keeping the WARC files to a consistent size just under the HDFS block size. For other collection formats, it is first necessary to create a zipped collection: a set of zipped files each containing an equal number of input documents in the required XML format.

The DiversityEngine provides a special mapper which allows processing of any WARC or zipped collection on the Hadoop cluster. A subset of the DiversityEngine analysis tools has been used to process the 1.8 million documents of the New York Times collection (from the HCIR 2010 challenge[5]) [44] as well as many other collections of comparable size [32]. In the case of the New York Times collection, the zipped collection of 1000 documents each corresponded to 1800 map jobs to be processed by Hadoop. Although indexing performance was not a primary concern in the platform implementation, the number of documents that can be processed within a time period scales horizontally with the number of available machines.

2.2 Evaluation Framework

The DiversityEngine is designed as a testbed for research into multimedia document analysis and an important aspect is the evaluation of techniques. To aid in this process, the DiversityEngine contains an evaluation framework that provides the means to easily evaluate the *quality* of a particular annotator. For this, a gold annotation (a perfect annotation) has to be provided to the framework. The evaluation process then executes the annotator pipeline and compares the generated annotations to the gold annotations. This comparison allows for computation of various standard, retrieval quality measures such as precision, recall and F-measure. Annotations are compared entity-by-entity and the process allows for any kind of entity comparison and quality measure.

While the evaluation framework is designed to measure annotation quality, it is not designed to measure the *runtime performance* of annotators, or measure the performance of applications that build on top of the DiversityEngine (such as search applications). The DiversityEngine provides execution times of analysers but this is not part of the evaluation framework.

Evaluation jobs are configured, as with all the DiversityEngine's jobs, using an XML configuration script. The script provides information about the location of the gold annotations, the location of the output annotations and information about which annotations to compare and how to compare them. The evaluation framework provides a set of evaluators and entity comparators which can be specified in the configuration, or the user may write their own to provide comparisons between more unusual entities.

A default evaluator counts matches between gold annotations and generated annotations and produces a

[3] The DiverstyEngine documentation contains speed estimates for each analysis tool

[4] http://hadoop.apache.org

[5] https://sites.google.com/site/hcirworkshop/hcir-2010/challenge

report of true positives, false positives and false negatives from which it is able to provide precision, recall and F1 measures.

This output is provided as an XML document which gives information about the experiment run, the counts for each of the documents encountered and the totals which include the precision and recall.

While evaluating the correctness of the top-most annotation in an annotation hierarchy, it would first be prudent to ensure that the annotations on which the higher-level annotation has been based, were also all evaluated as correct. The evaluation framework provides an evaluator (The Valid Parent Evaluator) which deems an annotation as correct only if all annotations at lower levels in the tree are also correct.

Both the default annotation and the valid parent annotator rely on external means to determine whether two annotations are equal. The framework provides a means for implementing custom entity matching algorithms. However, it also provides four basic algorithms which offer a range of equality measures to meet most needs.

To compare XML payloads, the framework uses XML comparators. The default comparator simply compares all nodes in the tree and ensures a complete tree match. A second comparator provides a means for comparing the tree while ignoring attribute names and values; some analysers, such as the face analysers, output the training model used to find faces in an attribute which has no bearing on how correct the face annotation is.

To run an evaluation, two datasets are required: the evaluation dataset and the golden dataset. Both datasets contain annotations of a single document corpus. The golden annotation dataset must contain correct data annotations. These can either be created by running the DiversityEngine over the document corpus and correcting the annotator mistakes manually, by conversion of some other format into the DiversityEngine's XML interchange format, or completely manually. The datasets must contain all the annotations that are required to make the annotation tree complete but does not need to include the original documents.

3 Extracting Facts and Opinions

The fundamental basis of any research into opinion analysis, bias and diversity in multimedia documents is a robust set of fact and opinion extraction routines for the text. The DiversityEngine provides state-of-the-art modules that can be used for entity and syntax extraction as well as opinion and sentiment extraction in text. Images are naturally more ambiguous when it comes to opinions and sentiments and are considerably harder to mine for entities. However, the DiversityEngine contains advanced visual analysis modules for supporting the extraction of facts and opinions from multimedia documents. Both sets of analysers are described briefly in the sections below.

3.1 Extraction of Facts and Opinions from Text

The first step in mining opinions from text is to identify statements and the syntactic relationship between them. To do this the DiversityEngine provides a number of new and pre-existing text analysis modules. The OpenNLP tool[6] splits the raw text into sentences and tokens and assigns a part-of-speech (POS) tag to each token. Named entities and coarse-grained word sense tags are extracted using the SuperSense tagger [12] while grammatical and shallow semantic structures [37] are extracted by the LTH-SRL tool [27]. A module based on the TARSQI Toolkit[7] has been developed for annotating temporal expressions in the document with annotations in the TimeML format[8].

We have applied the LTH-SRL parser in order to extract these structures[27,23]. In addition to the predicates and arguments, the tool outputs *grammatical* relations such as subject and object. The tool was built using the PropBank [37] and NomBank [33] databases, and achieved the top score in an international evaluation of predicate-argument extraction systems [47].

Figure 3 shows an example of the structure output by the LTH-SRL tool: The sentence *HRW denounced the defenceless situation of these prisoners* has been annotated with grammatical relations (above the text) and predicate-argument links (below the text). The predicate *denounced*, which is an instance of the PropBank frame `denounce.01`, has two semantic arguments: the Speaker (A0) and the Subject (A1, or Theme), which are realized on the grammatical level as a subject and a direct object, respectively. Similarly, *situation* has the NomBank frame `situation.01` and an Experiencer semantic argument (A0).

The DiversityEngine also has access to the AIDA service [53] which, given some text, extracts and disambiguates entities and links them to open-linked-data re-

[6] `http://opennlp.sourceforge.net`

[7] http://www.timeml.org/site/tarsqi/

[8] http://www.timeml.org/site/index.html

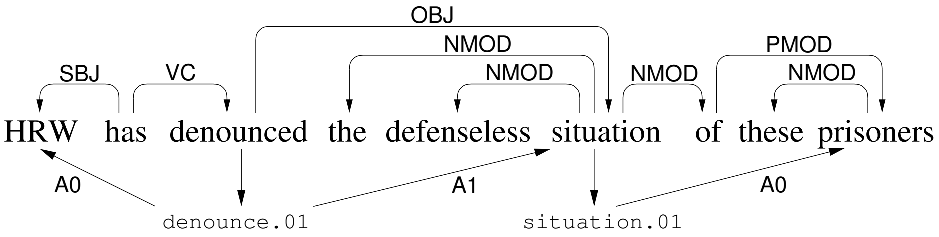

OBJ
NMOD
PMOD
SBJ
VC
NMOD
NMOD
NMOD
HRW has denounced the defenseless situation of these prisoners
A0
A1
A0
denounce.01
situation.01

Fig. 3: Example sentence and its syntactic and shallow-semantic analysis.

sources such as YAGO, Freebase or DPpedia. The module in the DiversityEngine provides the disambiguated tokens and links them with some information about the entities.

Automatic systems for the analysis of opinions expressed in text on the web have been studied extensively during recent years [38]. Initially, this was formulated as a coarse-grained task – locating documents or passages expressing opinion – and was usually tackled using methods derived from standard retrieval or text categorisation techniques. However, in recent years there has been a shift towards a more complex task: not only finding the piece of text expressing the opinion, but also *who* holds the opinion and to *what* is it addressed; is it positive or negative (*polarity*); what is its *intensity*? These more complex problem formulations require us to move away from the simple categorisation-based methods and make use of a deeper analysis of linguistic structure.

Opinion extraction from text in the DiversityEngine consists of two parts: a very fast *coarse-grained* opinion extractor that finds text pieces (such as sentences) containing some expression of opinion and a *fine-grained* system that extracts detailed structural information about the opinions to support complex queries such as "give me all documents where Miliband expresses a negative opinion about Cameron."

To make it possible to analyse large quantities of text, the DiversityEngine's opinion analysis first applies a very fast classifier to quickly extract sentences containing some expression of opinion. As with earlier work on coarse-grained opinion analysis, opinionated sentences are classified to the category of subjective or objective using a binary classifier. This classifier is very fast and processes roughly 1,400 sentences per second. While the coarse-grained opinion classifier allows the detection of opinionated sentences in text, many applications require more detailed information about the opinion, such as identifying the entity holding the opinion and determining its polarity and intensity. This extraction process is carried out by the fine-grained opinion analysis module [25,26] which we implemented as a sequence of statistical systems trained on the MPQA dataset[51].

The module consists of several distinct stages. To extract opinion expressions, a standard sequence labeller is used for subjective expression markup similar to the approach in [7]. Once opinion expressions have been extracted, the module assigns features to every expression: the *opinion holder*, the *polarity* and the *intensity*. The problem of extracting opinion holders for a given opinion expression is in many ways similar to argument detection in predicate-argument structure analysis [10,42]. We therefore approached this problem using methods inspired by predicate-argument analysis, and trained support vector classifiers [6] that were applied to the noun phrases in the same sentence as the opinion expression.

Polarity and intensity features are assigned to every expression. The polarity feature takes the values Positive, Neutral, and Negative and the intensity feature Low, Medium, and High. We trained linear support vector machines to carry out these classifications. The problem of polarity classification has been studied in detail by Wilson et al. [52] who used a set of carefully devised linguistic features. The DiversityEngine's classifiers are simpler and based on fairly shallow features: words, POS tags and subjectivity clues extracted from a window around the expression under consideration. Further details of the opinion extraction and evaluation can be found in [25,26].

3.2 Extraction of Supporting Information from Images

Images are included in web documents for a variety of reasons. Typically they are used to emphasise a particular aspect of text. For example In a news report of a demonstration the anger of the demonstrators may be emphasised by including an appropriate image or the pleasure of the recipient of an Oscar emphasised by a close up of the beaming winner. Some analysis of images in documents can provide additional evidence for opinions extracted from the text and in some cases may provide evidence, not obvious from the text, of attempts to unfairly influence the response of the reader [55, 5]. A well known example was the augmenting of a war zone image to make the smoke and fire more intense than appeared in the original.

The latest techniques for photo-montage creation means that, to the eye, it can be impossible to determine if a photograph is a true representation of what really happened. Discovering the semantic information within an image derived from a photo-montage may highlight how the exploitation of a particular image in a communication process aims to polarize opinions, and may provide evidence that a biased view is being projected.

The DiversityEngine includes some modules which implement state-of-the-art forensic image analysis [40]. In particular, there are modules for determining important aspects of image history including the type of device used for producing the digital content and whether the image has suffered any tampering and, if so, what kind and in what parts.

One of the modules is used for detecting tampering of faces within images; for example, replacing one face with another or altering the expression on the depicted person's face. A face detector is applied to the image in order to locate the faces, then three different tampering detection techniques are used to analyse the extracted regions. The module uses various methods for detecting manipulation: detection of a digital composite by exploiting JPEG compression artefacts to detect cropped and re-compressed images which result in misaligned JPEG blocks [31]; detecting photo-montages by exploiting double quantisation effects within the discrete cosine transform (DCT) coefficients [29]; and detection of forgeries by finding DCT coefficients that were previously compressed with higher quantisations ("JPEG Ghosts").

Image forgery can be achieved in a number of ways and different kinds of forgeries leave different traces within the image. In order to determine the authenticity of a particular image (as opposed to the precise manipulation) an image forgery detection module has been developed which fuses the output from a number of tools to provide a single "authenticity score" for the image. The image (or region of the image) is analysed using state-of-the-art image forensics tools based on JPEG compression artifacts: two novel algorithms developed for the DiversityEngine, which check for grid misalignments [4] and for copy paste forgeries [3] as well as an implementation of an existing algorithm [17]. Fusing the results of these detection algorithms is non-trivial and a formalization has been developed [18] based on Dempster-Shafer's Theory of Evidence (DST) [45]. This module's performance is encouraging: for a fixed false alarm probability of 5%, the module yields a Probability of Detection of 95% on a synthetic dataset which drops to 70% on a more challenging real-world dataset.

The forensic modules described so far implement the current state-of-the-art in image forensic analysis which is solely based on the analysis of single images. The DiversityEngine contains a novel image dependencies module, which attempts to determine the dependencies between images in a group based on both their visual and forensic links [41]. As well as having a role in the understanding of the opinion forming process, this analysis also lends itself to determining copyright infringement in the reuse of imagery. The module supposes that an image B depends on an image A if B is a modified version of A and that the modifications are either geometric (scaling, rotation, cropping), colour manipulation (colour transfer, brightness and contrast correction) or JPEG compression. It is also assumed that after any processing step the manipulated image is always JPEG compressed. The module estimates the modification parameters between two images' visual content to calculate a similarity score. Then a correlation score is calculated using the image processing noise extracted using a Wavelet-based de-noise algorithm. From all pairwise comparisons within the image group, a dependency graph is built where edges exist between vertices representing images if the correlation score is above a threshold. The dependency graph is pruned using a set of rules (an ontology); for example if one image depends on another then vice-versa is not possible, or that an image can have an arbitrary number of children but only one father (i.e. no photo-montages are allowed).

The extraction of high-level (semantic) information from images is markedly more complex than similar extraction from text since the information within images is significantly less explicit. This means that such extraction is somewhat less robust, except in some constrained cases. Extraction of sentiment from images is further complicated by the subjectivity of visual cues.

However, when used together, text and images can introduce some predictability and therefore strengthens the information extraction obtainable from the text alone. It is for these reasons and the particular value of forensic analysis that the DiversityEngine provides image analysis modules for **supporting** the extraction of facts and opinions from text.

The following modules mainly focus on the annotation of images with useful tags; that is, tags representing image content or sentiment. Some modules provide basic building blocks from which more advanced image analysis modules can be built. Many of the image feature extraction techniques in the modules use OpenIMAJ [21] as the underlying library for image analysis. Some of the modules perform basic binary image classification such as the Man-made/Natural classification module which is based on an edge-direction coherence vector image feature [48]. More advanced indexing and analysis can be performed using an advanced bag of visual words approach; an implementation of a Difference-of-Gaussian SIFT [30] key-point detector is included in the DiversityEngine which efficiently finds and describes interest points within the image [22]. These interest points can be indexed and matched using ImageTerrier [20] module to allow efficient retrieval of similar images to a query. If the index images are tagged and the query is not, the DiversityEngine can perform automatic annotation using a nearest-neighbours technique; for example a module can automatically geo-tag untagged images based on visual similarity to matching (geotagged) landmarks [20].

The DiversityEngine includes a module which finds a score of image similarity between every image in a corpus. The module extracts SIFT features from all the images, indexes them using ImageTerrier then queries the index with each image in turn to find the most similar images in the corpus. Thresholding the resultant set allows the detection of image reuse in the corpus. Finding where an image has been reused provides a means for detecting trends in particular diversity metrics that can be extracted either from the images themselves or from the articles where the image has been reused. For example, images can be plotted against time to find when particular images were important within the corpus which, in turn, exposes an implied narrative for an image. It is also possible to plot the images against some other axis of diversity. Figure 4 shows image reuse plotted against an opinion value which is essentially the average subjectivity of the subjective expressions within the associated articles. It shows how, in our corpus, some images tend to be used for negatively polarised articles, where as others are used in a balanced way (note the strong vertical line represents the neutral position).

There is also a module for annotating images based on the dominant set clustering (DSC) algorithm [39]. This algorithm recursively finds dominant sets in the training data similarity graph to automatically generate a set of clusters against which new images can be compared. The DSC method has the advantage that the number of clusters does not need to be set *a priori.* The module in the DiversityEngine uses the MPEG-7 visual descriptors [46] to compute image similarity and associates the clusters with the tags from the images within. A new image is annotated by associating it with the tags of the first clusters to which it best matches. A decision algorithm based on the computation of the mean square error (MSE) is used, where for each cluster a feature vector that represents all the images in that cluster is derived (e.g., the average of all the feature vectors). The tags of the three clusters with the smallest MSE are used for the annotation. Each tag is the common tag of the images belonging to each cluster. In tests, this module returns an average accuracy of 73% considering the lowest MSE tag. The accuracy increases up to 80% when the second cluster is considered (so as at least one of the first two tags is correct), and up to 84% when the third cluster is considered as well (so at least one of the first three tags is correct). The accuracy depends on the considered database and could be improved when the annotation tool is jointly used with additional techniques based on text analysis and/or domain information.

As well as treating images as whole scenes, the DiversityEngine includes modules that attempt to extract important subsections of the image. There are two modules for extracting faces from images. The first [43] projects an image into hue-space and uses a segment of the hue dimension to define skin-like colours. It then discards connected-components of the image based on size, texture and shape until only face-shaped, skin-toned areas remain. The second uses the ubiquitous Haar-cascade implementation of Viola and Jones [49] to find faces in the image. Both face detection modules provide bounding boxes for the detected facial region.

More generally, the DiversityEngine includes a module for detecting the visually salient regions within an image; that is, region which are likely to contain the important objects of an image. Finding salient regions is a current research topic and the module in the DiversityEngine uses a novel integration of visual saliency and segmentation algorithms which attempts to provide more useful salient region maps than the current state-of-the-art [36]. The module takes a number of cues from the image to determine saliency, including the ranked sparse primary colours from the CIELab space as suggested by [16], the segmentation of the image using EDISON [11], global colours, local colours, and local luminance contrast. The segment features are then classified using a trained naïve Bayes classifier which gives the probability of each segment being salient. Experi-

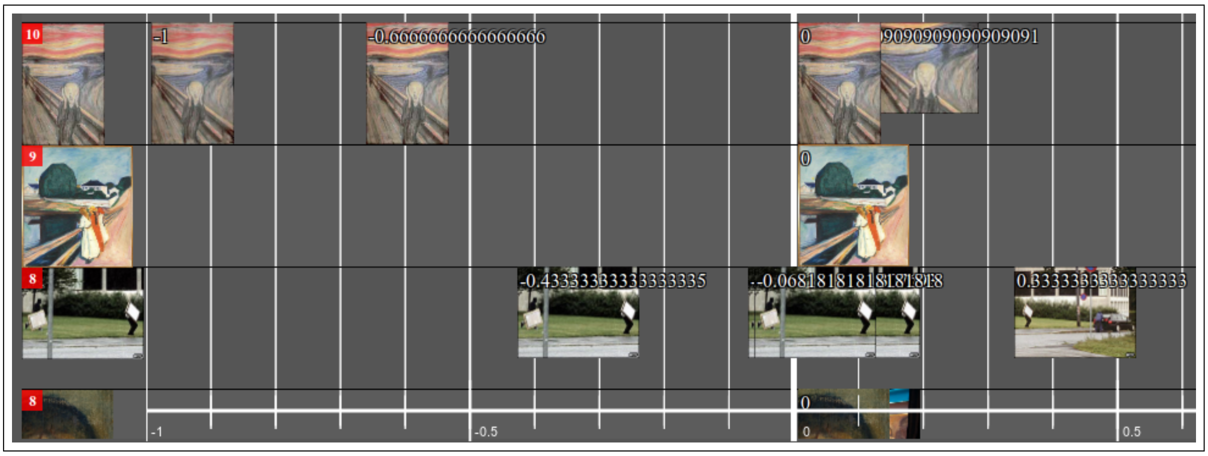


Fig. 4: Plotting image reuse against article opinion. The opinion value ranges from -1 (negative polarity) to +1 (positive polarity), where 0 is neutral (or no subjective expressions could be detected in the text with which the article is associated). A representative image is shown in the left column and the reused images plotted across opinion. Notice how some of the reused images are sub-images.

mental tests demonstrate that the saliency maps generated by the tool successfully highlight the main region of objects of interest. Figure 5 shows some sample images and their saliency maps. In the maps, the lighter the regions the more salient they are. Further details are available in [36].

3.3 Document Layout

One of the shortcomings of using text or image analysis alone is that the links between visual content and text content are not exploited. It is easy to manually pick out all the images or text from a webpage but without any information about the whereabouts of these elements, it is hard to make assertions about relationships between them. The DiversityEngine provides two different tools for the extraction of document layout information.

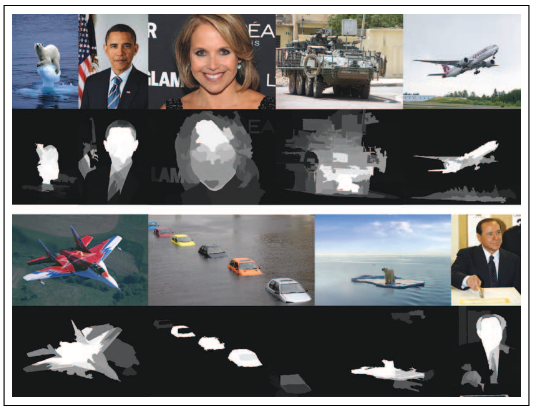


Fig. 5: Examples of original images and corresponding saliency maps.

The first is used as part of the core DiversityEngine platform. The Readability4J library[9] was developed as part of the DiverstityEngine project to determine the important article text within a webpage, effectively removing the page template, navigation and other spurious artefacts. It uses various voting heuristics to score HTML DOM (Document Object Model) elements within a web page structure based on density of text and class and identifier names and propagates these through the DOM tree. In most cases this process identifies the main text content of the article. The library is then able to use this information to extract the headline, author, publication date, article sub-headings, images associated with the article (as opposed to decoration or spurious images) and links from the article text. The library also provides a few additional functionalities that make it possible, with some other modules within the DiversityEngine, to track back to the absolute character within the HTML code, the annotations made with the DiversityEngine's modules.

The second tool providing complementary annotations is a module which extracts information about a specific rendered HTML layout. Due to the stylisation rules applied during the rendering of HTML, the distance between any two elements within the HTML DOM tree is not necessarily correlated to the distance when rendered to the screen. To expose this information, the HTML Layout Extractor module renders a web-page

[9] Now available as OpenSource through the OpenIMAJ project at `http://openimaj.org/`

into a fixed size off-screen buffer and measures the size and location of all the rendered elements. This information is then written to an annotation file which can be used to determine absolute rendered distances between elements. It is important that the measured page is rendered at a specific and fixed size, as it is often the case that web-pages scale to fit the render viewport which would make comparison between pages impossible.

4 Search Engine and API

For indexing and searching, The DiversityEngine uses Apache Solr [2], a popular open source search engine. Because Solr has a well established plug-in framework, it provides the DiversityEngine with a means for enhancing the search results as well as allowing users to easily build upon the DiversityEngine search API.

As most of the analysis tools work by enriching documents with metadata, we can map this metadata to the fields of a Solr document to provide search over the analysis results. For example, a named-entity recogniser will identify the *person* entities in a document and these can be mapped to a Solr field called *person*. Using the Solr API, this additional field allows for more expressive queries than a simple keyword query: a query can be formulated to retrieve documents that contain a particular person or, using the Solr faceting functionality, to request the person entities most frequently occurring in documents that match a given query.

The key step in configuring this within the DiversityEngine platform is indicating how the analysis results map to the Solr fields in a document. This is controlled by a Solr conversion XML file which maps an annotation entity type to a Solr field. The conversion file allows for specification of direct annotation indexing (also through XPath selection) as well as using filter classes for Solr field generation. For example, the ‘facts’ analyser uses the YAGO ontology to extract named entities from the article text. It outputs XML similar to that shown in Listing 1. The annotation maps entities (it provides *YAGO-entities*) to character positions in the text and then gives further information (it provides *facts*) about the entities in another annotation. Those entities can be converted into Solr fields using a simple conversion file. Listing 2 shows a snippet of this file. The two lines map two different Solr fields. The first is a basic copy field which uses the extracted entity name as the field value; the second takes the name of those entities which are associated with a specific WordNet category (countries) as the field value.

Listing 1: YAGO fact extractor annotation

```
<?xml version="1.0" encoding="UTF-8"?>
<lk-annotation>
  <annotation scope="...00000.arc.15521713.lktext.xml
      "
        provides="yago-entities">
    <e id="1" start="#649" end="#668">
        Federal_Constitutional_Court_of_Germany</e>
    <e id="2" start="#151" end="#158">Slovakia</e>
    <e id="3" start="#638" end="#647">The_German</e>
    ...
  </annotation>
  <annotation provides="facts">
    ...
    <e id="26" on="#2">
      <entity-information>
        <id>Slovakia</id>
        <indices>
          <index start="#151" end="#158" />
        </indices>
        <facts>
          <hasPopulationDensity>111#/km^2
                </hasPopulationDensity>
        ...
          <type>wordnet_country_108544813</type>
          <type>wordnet_district_108552138</type>
        ....
        </facts>
      </entity-information>
    </e>
  </annotation>
</lk-annotation>
```

Listing 2: Basic Solr conversion definition

```
<field solr="yago" annotation="yago-entities"
        value="\$text" />

<field solr="yago-country" annotation="facts"
        value="xpath:/entity-information[facts/type/
             text()='wordnet_country_108544813']/id/
             text()" />
```

By default, there is a one-to-one mapping between a DiversityEngine document and a Solr document. However, in some cases, particularly when indexing opinions, it can be advantageous to index at a more granular level, such as at sentence-level. For example, it is possible that many documents will contain both positive and negative statements and while it is possible to aggregate this information and assign an overall *polarity* score for a document, it may be more advantageous to index at a sentence level where there will be less ambiguity. Then it will be possible to search for sentences that match a particular keyword query that also contain *positive* or *negative* expressions. As with named entities, it is also possible to aggregate the number of positive and negative expressions contained in documents that match a given query, possibly indicating the overall polarity for a particular keyword query. The DiversityEngine provides extensions to Solr that allow for efficiently computing how documents matching a particular query change over time. This functionality was used extensively in the Yahoo Time Explorer application (see Section 5.2) to generate timelines. These timelines show both how the frequency of documents discussing a particular topic varies over time and also how the entities contained in those document change over time.

4.1 Search Result Diversification

Result diversification aims at minimising the risk of dissatisfaction by balancing the relevance and the novelty of the search results. Diversification of search results on unstructured documents is a well-studied problem (see e.g. [8,13,19,50]) but the annotated data offered by the DiversityEngine provides another interesting target for diversification. The key challenge here is to give a diverse set of results: an overview of the major plausible interpretations of a keyword query in terms of the document content and metadata. This effectively reduces the search space towards the intended search results. It could also provide a better understanding of the search space giving the opportunity to explore other parts of the space.

As part of the DiversityEngine's extensible search framework, various diversification algorithms can be seamlessly auditioned. Indeed, the DiversityEngine provides a general API to customise search result diversification. A state-of-the-art incremental diversification algorithm [34] is included that enables efficient processing of streaming search results over large-scale data.

In order to calculate the novelty of a document, typical diversification algorithms compute the distance between all of the search results using some distance metric and the API of the DiversityEngine enables customisation of this similarity function. Typically, the distance among unstructured documents is computed based on the whole document content, using e.g. cosine similarity [8] or using document fragments [19]. To leverage the named-entity annotations provided by the DiversityEngine, we have defined an entity-based similarity function.

Entity-based similarity metrics enable efficient diversification of search results based on their annotations because the features that are compared are smaller than those used for unstructured document comparisons. The similarity between one search result and another is computed as an overlap of the named-entities (e.g. people and/or locations) associated with the results. The entity-based similarity between two annotated documents, d_i, and d_j, is defined as the Jaccard coefficient between the two sets of entities, E_{d_i} and E_{d_j}, associated with these documents as shown in Equation 1.

$$Sim(d_i, d_j) = \frac{|E_{d_i} \cap E_{d_j}|}{|E_{d_i} \cup E_{d_j}|}. \tag{1}$$

The DiverstityEngine enables a standardised evaluation of search result diversification through its evaluation framework. The input to the evaluation procedure is a set of queries and their corresponding ground truths. The evaluation program executes the queries and performs search result diversification using the algorithms and distance functions specified by the application. The output of the evaluation is a matrix containing the scores for each query and an evaluation measure in the standardised format as specified by TREC.

The DiverstityEngine integrates several state-of-the-art measures of diversification quality as offered by the TREC 2010 Web Track[10]. The supported evaluation measures include: Expected Reciprocal Rank (ERR) [9], alpha-DCG and alpha-nDCG [13], Novelty- and Rank-Biased Precision (NRBP) [14], intent aware mean average precision (MAP-IA) [1] and subtopic recall [54].

5 Applications of the Software

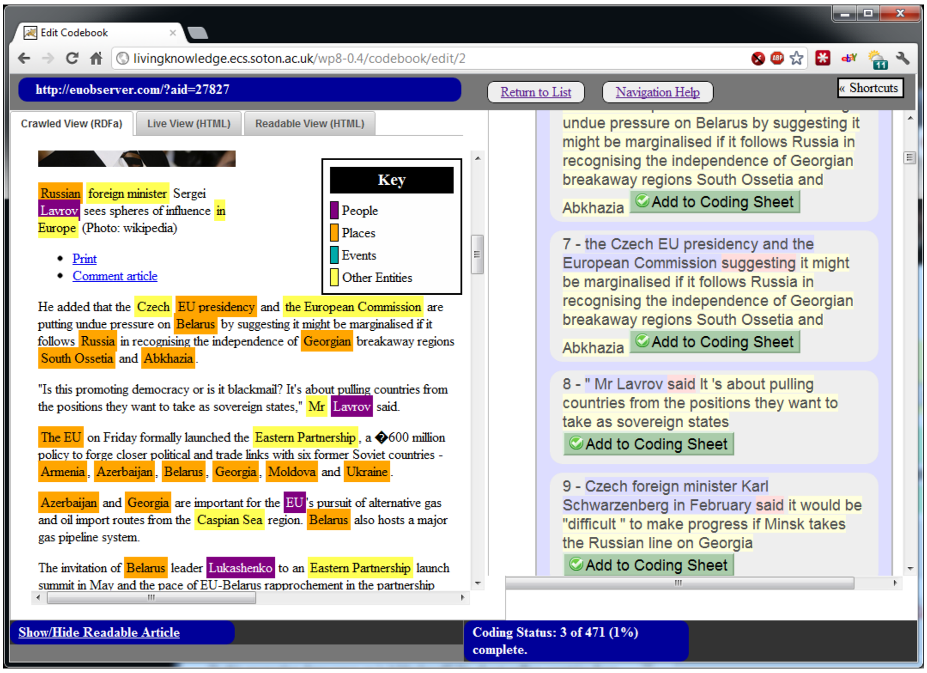
As part of the LivingKnowledge project, the DiversityEngine was used to produce two very distinct applications requiring the use of document mining: the Media Content Analysis application and the Time Explorer.

5.1 Media Content Analysis

SORA is a social research and consulting company based in Vienna whose main product is their expertise in the analysis of news and broadcast media content to generate statistics and trends. Much of their work involves the hand annotation of media articles by employees known as Coders. The coders read selected articles and fill in a *coding sheet* by hand for each article. The coding sheet captures the information for statisticians to produce the analysis required by the customer. This could be, for example an analysis of positive and negative references to a political party in a particular period in the press.

It is clear that such a manual process takes considerable effort and hence time and money. Natural language

[10] http://trec.nist.gov/data/web10.html



Russian foreign minister Sergei Lavrov sees spheres of influence in Europe (Photo: wikipedia)

- Print
- Comment article

He added that the Czech EU presidency and the European Commission are putting undue pressure on Belarus by suggesting it might be marginalised if it follows Russia in recognising the independence of Georgian breakaway regions South Ossetia and Abkhazia.

"Is this promoting democracy or is it blackmail? It's about pulling countries from the positions they want to take as sovereign states," Mr Lavrov said.

The EU on Friday formally launched the Eastern Partnership, a �600 million policy to forge closer political and trade links with six former Soviet countries - Armenia, Azerbaijan, Belarus, Georgia, Moldova and Ukraine.

Azerbaijan and Georgia are important for the EU's pursuit of alternative gas and oil import routes from the Caspian Sea region. Belarus also hosts a major gas pipeline system.

The invitation of Belarus leader Lukashenko to an Eastern Partnership launch summit in May and the pace of EU-Belarus rapprochement in the partnership

undue pressure on Belarus by suggesting it might be marginalised if it follows Russia in recognising the independence of Georgian breakaway regions South Ossetia and Abkhazia ✓Add to Coding Sheet

7 - the Czech EU presidency and the European Commission suggesting it might be marginalised if it follows Russia in recognising the independence of Georgian breakaway regions South Ossetia and Abkhazia ✓Add to Coding Sheet

8 - " Mr Lavrov said It 's about pulling countries from the positions they want to take as sovereign states ✓Add to Coding Sheet

9 - Czech foreign minister Karl Schwarzenberg in February said it would be "difficult " to make progress if Minsk takes the Russian line on Georgia ✓Add to Coding Sheet

Coding Status: 3 of 471 (1%) complete.

Fig. 6: The media content analyser application, built using the DiversityEngine. The left side shows the article being coded and the automatically highlighted entities extracted. The right side shows part of the coding sheet and some of the automatically extracted subjective sentences which are used to fill the coding sheet.

processing tools, like the DiversityEngine, can automatically do some of the work for the coders saving time and money. To enable this, the DiversityEngine platform was used as a basis for a web-based application that implements a coding sheet and uses DiversityEngine tools for extracting some of the coding sheet results automatically.

So that the application was flexible and extensible, the codebook (the definition of the fields of the coding sheet) was created using a straightforward XML schema which is used for delivering the various parts of the application. A new module was implemented as a DiversityEngine analyser which aggregates the results from many of the DiversityEngine analysis tools and creates a coding sheet with some of the fields automatically populated.

When a user logs in to the web-based coding interface (shown in Figure 6) and begins to code a new article, the fields that were automatically generated are entered automatically to populate the coding sheet where appropriate. and the web-application provides the coder with a quick way to accept or reject the results of the DiversityEngine's analysis. Examples of the types of information that the DiversityEngine can automatically provide include extraction of people's names and faces from the text and images, extraction of locations and events (time-based entities) from the text, indications of whether images have been manipulated and automatic extraction of claims (subjective sentences where entities assert opinions). The DiversityEngine also provides other less challenging but useful analyses such as extraction of the article date and headline, an indication as to whether the page contains advertisements and the number of words in the article (previously estimated manually by SORA's coders). User trials in the company demonstrated a substantial speed up in the time to complete the coding sheets without losing accuracy and level of detail.

5.2 Time Explorer

The Time Explorer [32][11] is a web application developed by Yahoo! research that aims to make the diversity of a corpus an asset in search and retrieval applications. The goal is to provide tools that allow the exploration of web articles from many points of view and crucially to reveal how the knowledge within them evolves over time. The application is built entirely on top of the DiversityEngine. Ultimately, the aim is to include many aspects of diversity although initially the application focuses only on the time dimension.

[11] http://fbmya01.barcelonamedia.org:8080/future/

The Time Explorer uses analysis tools available in the DiversityEngine to extract from each document the persons, locations and organisations as well as all the time expressions that can be resolved to a specific day, month or year. Time expressions are extracted from both explicit references (as in "September 2010") and relative references as in "next month". In addition, simple heuristics are used to assign a subset of the entities as keywords for the document. From these features, two indexes are created: one for each document in the collection and one for each sentence in the collection. For the sentence-level index, a content date is computed as one or more of the event dates found in the document or, if there are no event dates, the publication date is used. For example, given the following hypothetical document with publication date in May 1^{st}, 1999, two sentences will be found:

> George Bush was elected governor of Texas in 1994. George Bush will run for President of the United States next year.

George Bush will be extracted as a person mention in both sentences. *Texas* will be extracted as a location mention in the first sentence and *United States* will be extracted as a location mention in the second sentence. *1994* will be extracted as a time expression in the first sentence and *next year* will be extracted as a time expression in the second sentence and resolved to 2000. The content date for the first sentence will be May 1^{st}, 1999 whlile the content date of the second sentence will be 2000.

The resulting indexes allow for a wide range of queries including, for example:

- Returning the documents that contain specific phrases and/or entities, and chronological counts thereof,
- Returning a list of people most related to the a specific entity (place, person, organisation, etc.),
- Returning documents that contain specific phrases and/or entities that were published during a particular period of time; and
- Returning documents that mention events from a specific time (not necessarily related to the published date).

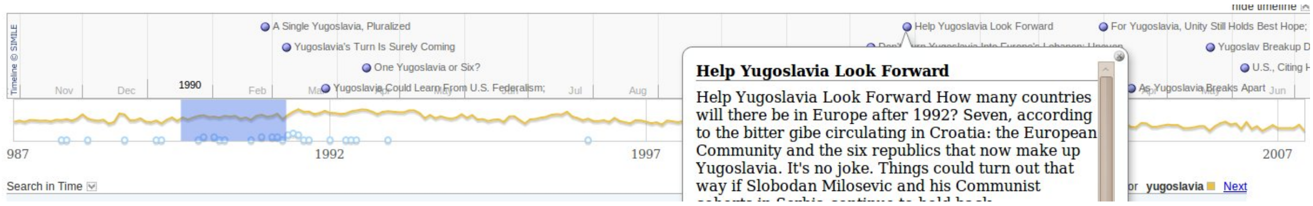
These queries and combinations of them are very powerful, so much so that it is unlikely that a user will be able to express the queries in a meaningful way. Therefore defining an intuitive user interface is extremely important.

The focus of this application is to aid the understanding of how topics evolve over time and so, unsurprisingly, the core of the user interface is a timeline. Though there are many timelines available, including Google Trends, Google Timeline and many derived from the Simile Timeline widget [28], the Time Explorer attempts to improve on these implementations by combining many of the best features of each. Figure 7 displays the timeline produced for the query "Yugoslavia". The timeline data is split between two bands. The bottom band is the trend graph and displays how the frequency of documents containing the term Yugoslavia changes over the considered time period (Figure 7 shows the whole 20 years covered by the New York Times Annotated Collection [44] which the application demo uses); while the top band, the topic timeline, displays the titles of the top-ranked articles. The user can click on the title of articles to get a document summary and they can also jump directly to the article. Both timeline bands are interactive and can be scrolled to change the considered time period. For large corpora, there is a trade-off between response time and coverage on the timeline. So more documents can be retrieved for a particular time period by triggering a search by simply clicking in the highlighted region for the considered time period.

In addition to discovering the articles, an entity filter panel displays the entities most associated with the query. In this case, entities are people and locations extracted by DiversityEngine tools and those that were originally annotated in the corpus. Selecting an entity will submit a query for all documents that mention (or do not mention) that entity. It also provides links to see a definition of the entity if one can be found in Wikipedia.

An additional feature of the entity filter is to provide a trend line for the entity on the trend graph. This displays the entity's mentions in the timeline in order to allow comparison and see correlations between the entity mentions and the query results. Figure 8 shows this feature for two entities (*President Clinton* and *General Wesley Clark*) drawn in red and green on the trend graph.

The entity list and the entity co-reference graph widget are both modified in real-time as the query is refined or the timeline period is changed. This allows the user to easily see how the important entities change over time for a given query. For example, for the query Yugoslavia many sports figures were associated with the term before the Yugoslavian conflict in 1990 but from then on the names of world leaders become much highly more correlated.


hide timeline
Timeline © SIMILE
A Single Yugoslavia, Pluralized
Yugoslavia's Turn Is Surely Coming
One Yugoslavia or Six?
Yugoslavia Could Learn From U.S. Federalism;
Help Yugoslavia Look Forward
For Yugoslavia, Unity Still Holds Best Hope;
Yugoslav Breakup D
U.S., Citing H
As Yugoslavia Breaks Apart
Nov
Dec
1990
Feb
Ma
Jul
Aug
Jun
987
1992
1997
2007
Search in Time
or yugoslavia
Next

Help Yugoslavia Look Forward

Help Yugoslavia Look Forward How many countries will there be in Europe after 1992? Seven, according to the bitter gibe circulating in Croatia: the European Community and the six republics that now make up Yugoslavia. It's no joke. Things could turn out that way if Slobodan Milosevic and his Communist

Fig. 7: The Time Explorer's timeline control and the view of a document summary

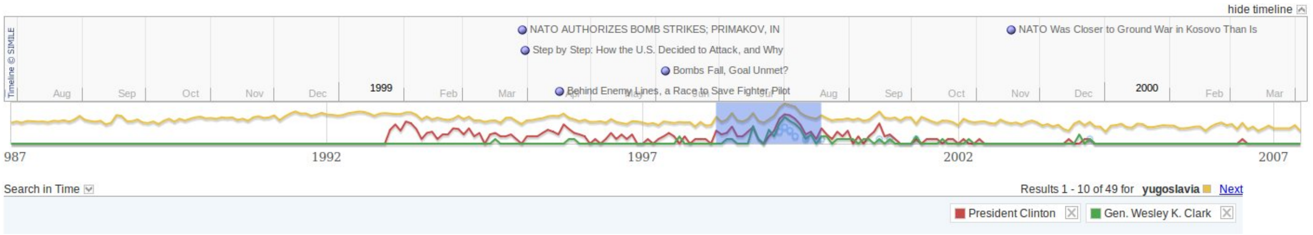

hide timeline
Timeline © SIMILE
NATO AUTHORIZES BOMB STRIKES; PRIMAKOV, IN
Step by Step: How the U.S. Decided to Attack, and Why
Bombs Fall, Goal Unmet?
Behind Enemy Lines, a Race to Save Fighter Pilot
NATO Was Closer to Ground War in Kosovo Than Is
Aug
Sep
Oct
Nov
Dec
1999
Feb
Mar
Aug
Sep
Oct
Nov
Dec
2000
Feb
Mar
987
1992
1997
2002
2007
Search in Time
Results 1 - 10 of 49 for yugoslavia
Next
President Clinton
Gen. Wesley K. Clark

Fig. 8: The Time Explorer showing entity mentions over time

So far, the examples presented for the Time Explorer have been using the published date of the article, however it is also possible to use the content date for plotting events on the timeline. This has the advantage that events can be plotted into the future making it possible to see predictions in article texts. Using the content date, it is also possible to look for articles making predictions about a specific date that were made in the past.

The Time Explorer has many other features on top of those briefly described here and initial feedback from trials has been very favourable. Lists of both the most important keywords associated with the document and the most important dates are available. These serve both to better summarise the document and to provide an additional mechanism for refining the search. In addition, clicking on the source gives details about the source of the article. In the New York Times collection, this is obviously limited to the New York Times articles, but the Time Explorer has been extended to include daily updated news feeds from other media outlets such as The Washington Post, The Guardian, The BBC and also blogs. The Time Explorer also includes tag clouds and maps for exploring important terms and locations in query results.

6 Conclusion

In this paper we have described the open-source DiversityEngine platform and testbed with which multimedia analysis and retrieval can be performed. An overview of the range of novel analysis modules delivered with the platform has been presented together with references to the more detailed papers on the underlying algorithms and their evaluation. Performance of the search and retrieval engine depends on the specific platform used for the deployment of Solr. The platform also allows the multimedia feature analysis and indexing to be scaled horizontally using Hadoop. The engine makes the integration of analysis modules as simple and transparent as possible, while providing flexible and extensible higher-level APIs on which applications can be built. It is available for experimentation, as it comes with some state-of-the-art text and image analysis algorithms including article metadata extraction, syntactic text extraction, opinion extraction from text, forensic analysis of images and image similarity. All are included either as open-source or as binary modules in the DiversityEngine.

We have described two applications of the platform that show how it can be used to forward the current state of the art in media analysis, search and retrieval. The Time Explorer was been released as a demo in Yahoo!'s sandbox and the Media Content Analyser is being taken up by SORA as part of its workflow; both applications demonstrate that the DiversityEngine is a versatile, state-of-the-art platform for fact, opinion and diversity analysis for multimedia retrieval with the additional advantage of being open-source.

Acknowledgements This work was supported by the European Union under the Seventh Framework project LivingKnowledge (IST-FP7- 231126). We would also like to thank all our partners who contributed to the LivingKnowledge project on which this work has been built.

References

1. Agrawal, R., Gollapudi, S., Halverson, A., Ieong, S.: Diversifying search results. In: WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining, pp. 5–14. ACM, New York, NY, USA (2009). DOI http://doi.acm.org/10.1145/1498759.1498766
2. Apache Software Foundation: Solr. `http://lucene.apache.org/solr/` (2010)
3. Bianchi, T., De Rosa, A., Piva, A.: Improved dct coefficient analysis for forgery localization in jpeg images. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 2444–2447. IEEE (2011)
4. Bianchi, T., Piva, A.: Detection of non-aligned double jpeg compression with estimation of primary compression parameters. In: IEEE International Conference on Image Processing (2011)
5. Boato, G., De Natale, F., Zontone, P.: How digital forensics may help assessing the perceptual impact of image formation and manipulation. In: Video Processing and Quality Metrics for Consumer Electronics (2010)
6. Boser, B., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152. Pittsburgh, United States (1992)
7. Breck, E., Choi, Y., Cardie, C.: Identifying expressions of opinion in context. In: IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 2683–2688. Hyderabad, India (2007)
8. Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, pp. 335–336. ACM, New York, NY, USA (1998). DOI http://doi.acm.org/10.1145/290941.291025
9. Chapelle, O., Metlzer, D., Zhang, Y., Grinspan, P.: Expected reciprocal rank for graded relevance. In: Proceedings of the 18th ACM conference on Information and knowledge management, CIKM '09, pp. 621–630. ACM, New York, NY, USA (2009). DOI 10.1145/1645953.1646033. URL `http://doi.acm.org/10.1145/1645953.1646033`
10. Choi, Y., Breck, E., Cardie, C.: Joint extraction of entities and relations for opinion recognition. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, pp. 431–439. Sydney, Australia (2006)
11. Christoudias, C., Georgescu, B., Meer, P.: Synergism in low level vision. In: Pattern Recognition, 2002. Proceedings. 16th International Conference on, vol. 4, pp. 150 – 155 vol.4 (2002). DOI 10.1109/ICPR.2002.1047421
12. Ciaramita, M., Altun, Y.: Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In: Processings of the 2006 Conference on Empirical Methods in Natural Language Processing, pp. 594–602. Sydney, Australia (2006)
13. Clarke, C.L., Kolla, M., Cormack, G.V., Vechtomova, O., Ashkan, A., Büttcher, S., MacKinnon, I.: Novelty and diversity in information retrieval evaluation. In: SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, pp. 659–666. ACM, New York, NY, USA (2008). DOI http://doi.acm.org/10.1145/1390334.1390446
14. Clarke, C.L., Kolla, M., Vechtomova, O.: An effectiveness measure for ambiguous and underspecified queries. In: Proceedings of the 2nd International Conference on Theory of Information Retrieval: Advances in Information Retrieval Theory, ICTIR '09, pp. 188–199. Springer-Verlag, Berlin, Heidelberg (2009). DOI 10.1007/978-3-642-04417-5_17. URL `http://dx.doi.org/10.1007/978-3-642-04417-5_17`
15. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M.A., Saggion, H., Petrak, J., Li, Y., Peters, W.: Text Processing with GATE (Version 6) (2011). URL `http://tinyurl.com/gatebook`
16. Drelie Gelasca, E., Tomasic, D., Ebrahimi, T.: Which Colors Best Catch Your Eyes: a Subjective Study of Color Saliency. In: Fisrt International Workshop on Video Processing and Quality Metrics for Consumer Electronics, Scottsdale, Arizona, USA, ISCAS. SPIE (2005)
17. Farid, H.: Exposing digital forgeries from JPEG ghosts. IEEE Transactions on Information Forensics and Security **4**, 154–160 (2009)
18. Fontani, M., Bianchi, T., De Rosa, A., Piva, A., Barni, M.: A Desmpter-Shafer framework for decision fusion in image forensics. In: IEEE International Workshop on Information Forensics and Security (2011)
19. Gollapudi, S., Sharma, A.: An axiomatic approach for result diversification. In: WWW '09: Proceedings of the 18th international conference on World wide web, pp. 381–390. ACM, New York, NY, USA (2009). DOI http://doi.acm.org/10.1145/1526709.1526761
20. Hare, J., Samangooei, S., Dupplaw, D., Lewis, P.: Imageterrier: An extensible platform for scalable high-performance image retrieval. In: The ACM International Conference on Multimedia Retrieval (ICMR 2012) to appear (2012)
21. Hare, J.S., Samangooei, S., Dupplaw, D.P.: Openimaj and imageterrier: Java libraries and tools for scalable multimedia analysis and indexing of images. In: Proceedings of the 19th ACM international conference on Multimedia, MM '11, pp. 691–694. ACM, New York, NY, USA (2011). DOI 10.1145/2072298.2072421. URL `http://doi.acm.org/10.1145/2072298.2072421`
22. Hare, J.S., Samangooei, S., Lewis, P.H.: Efficient clustering and quantisation of sift features: exploiting characteristics of the sift descriptor and interest region detectors under image inversion. In: Proceedings of the 1st ACM International Conference on Multimedia Retrieval, ICMR '11, pp. 2:1–2:8. ACM, New York, NY, USA (2011). DOI 10.1145/1991996.1991998. URL `http://doi.acm.org/10.1145/1991996.1991998`
23. Johansson, R.: Statistical bistratal dependency parsing. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pp. 561–569. Singapore (2009)
24. Johansson, R., Moschitti, A.: A flexible representation of heterogeneous annotation data. In: Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10), pp. 3712–3715. Valetta, Malta (2010)
25. Johansson, R., Moschitti, A.: Reranking models in fine-grained opinion analysis. In: Proceedings of the 23rd International Conference of Computational Linguistics (Coling 2010), pp. 519–527. Beijing, China (2010)
26. Johansson, R., Moschitti, A.: Extracting opinion expressions and their polarities – exploration of pipelines and joint models. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 101–106. Portland, United States (2011)
27. Johansson, R., Nugues, P.: Dependency-based syntactic–semantic analysis with PropBank and NomBank. In: CoNLL 2008: Proceedings of the Twelfth Conference on

Natural Language Learning, pp. 183–187. Manchester, United Kingdom (2008)

28. Karger, D., Smith, M.: Simile timeline. http://www.simile-widgets.org/timeline/ (2009)
29. Lin, Z., He, J., Tang, X., Tang, C.: Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis. Pattern Recognition **42**(11), 2492–2501 (2009)
30. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision **60**(2), 91–110 (2004). DOI 10.1023/B:VISI.0000029664.99615.94. URL http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94
31. Luo, W., Qu, Z., Huang, J., Qiu, G.: A novel method for detecting cropped and recompressed image blocks. In: IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP) (2007)
32. Matthews, M., Tolchinsky, P., Mika, P., Blanco, R., Zaragoza, H.: Searching through time in the new york times categories and subject descriptors. Information Retrieval (2010)
33. Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., Grishman, R.: The NomBank project: An interim report. In: HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation, pp. 24–31. Boston, United States (2004)
34. Minack, E., Siberski, W., Nejdl, W.: Incremental diversification for very large sets: a streaming-based approach. In: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, SIGIR '11, pp. 585–594. ACM, New York, NY, USA (2011). DOI 10.1145/2009916.2009996. URL http://doi.acm.org/10.1145/2009916.2009996
35. Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Plale, B., Simmhan, Y., Stephan, E., den Bussche, J.V.: The open provenance model core specification (v1.1). Future Generation Computer Systems (2010). URL http://eprints.ecs.soton.ac.uk/21449/
36. Muratov, O., Zontone, P., Boato, G., De Natale, F.: A segment-based image saliency detection. In: IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP) (2011)
37. Palmer, M., Gildea, D., Kingsbury, P.: The proposition bank: An annotated corpus of semantic roles. Computational Linguistics **31**(1), 71–106 (2005)
38. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval **2**(1-2), 1–135 (2008)
39. Pavan, M., Pelillo, M.: A new graph-theoretic approach to clustering and segmentation (2003)
40. Rocha, A., Scheirer, W., Boult, T., Goldenstein, S.: Vision of the unseen: current trends and challenges in digital image and video forensics. ACM Computing Surveys (to appear)
41. Rosa, A.D., Uccheddu, F., Costanzo, A., Piva, A., Barni, M.: Exploring image dependencies: a new challenge in image forensics. In: Media Forensics and Security XII Conference, SPIE Electronic Imagining (2010)
42. Ruppenhofer, J., Somasundaran, S., Wiebe, J.: Finding the sources and targets of subjective expressions. In: Proceedings of the Sixth International Language Resources and Evaluation (LREC'08), pp. 2781–2788. Marrakech, Morocco (2008)
43. Sandeep, K., Rajagopalan, A.N.: Human face detection in cluttered color images using skin color, edge information. In: S. Chaudhuri, A. Zisserman, A.K. Jain, K.L. Majumder (eds.) ICVGIP. Allied Publishers Private Limited (2002)
44. Sandhaus, E.: The New York Times Annotated Corpus. LDC2008T19. Linguistic Data Consortium, University of Pennsylvania (2008)
45. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
46. Sikora, T.: The MPEG-7 visual standard for content description-an overview. Circuits and Systems for Video Technology, IEEE Transactions on **11**(6), 696–702 (2001). DOI 10.1109/76.927422. URL http://dx.doi.org/10.1109/76.927422
47. Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., Nivre, J.: The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In: Proceedings of CoNLL 2008, pp. 159–177. Manchester, United Kingdom (2008)
48. Vailaya, A., Jain, A., Zhang, H.: On image classification: city images vs. landscapes. Pattern Recognition **31**(12), 1921–1935 (1998). DOI 10.1016/S0031-3203(98)00079-X. URL http://dx.doi.org/10.1016/S0031-3203(98)00079-X
49. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features (2001). URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.3.7597
50. Wang, J., Zhu, J.: Portfolio theory of information retrieval. In: SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, pp. 115–122. ACM, New York, NY, USA (2009). DOI http://doi.acm.org/10.1145/1571941.1571963
51. Wiebe, J., Wilson, T., Cardie, C.: Annotating expressions of opinions and emotions in language. Language Resources and Evaluation **39**(2-3), 165–210 (2005)
52. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. Computational Linguistics **35**(3), 399–433 (2009)
53. Yosef, M.A., Hoffart, J., Bordino, I., Spaniol, M., Weikum, G.: AIDA: Accurate Online Disambiguation of Named Entities in Text and Tables. In: Proceedings of the 37th International Conference on Very Large Databases, VLDB 2011, pp. 1450–1453 (2011)
54. Zhai, C.X., Cohen, W.W., Lafferty, J.: Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, SIGIR '03, pp. 10–17. ACM, New York, NY, USA (2003). DOI 10.1145/860435.860440. URL http://doi.acm.org/10.1145/860435.860440
55. Zontone, P., Carli, M., Boato, G., De Natale, F.: Impact of contrast modification on human feeling: an objective and subjective assessment. In: IEEE Conference on Image Processing (ICIP) (2010)