Preprint, final version of the article is at http://doi.acm.org/10.1145/2507157.2508217

GAIN: web service for user tracking and preference learning – a SMART TV use case

Jaroslav Kuchař*
Web Engineering Group
Faculty of Information Technology, Czech Technical University, Prague
jaroslav.kuchar@fit.cvut.cz

Tomáš Kliegr†
Department of Information and Knowledge Engineering
University of Economics, Prague
tomas.kliegr@vse.cz

ABSTRACT

GAIN (`inbeat.eu`) is a web application and service for capturing and preprocessing user interactions with semantically described content. GAIN outputs a set of instances in tabular form suitable for further processing with generic machine-learning algorithms. GAIN is demoed as a component of a "SMART-TV" recommender system. Content is automatically described with DBpedia types using a Named Entity Recognition (NER) system. Interest is determined based on explicit user actions and user's attention computed by 3D head pose estimation. Preference rules are learnt with an association rule mining algorithm. These can be e.g. deployed to a business rules system, acting as a recommender.

Introduction

An important part of the personalization process is the capture of user interaction and preprocessing the recorded data to a form that can be handled by the preference learning algorithm of choice. GAIN[1] is a generic web application and web service for handling this task. It was originally developed to supplement conventional clickstream analysis, however, it can be used also in other domains, such as for tracking "SmartTV" users as featured in this paper.

In this usecase, GAIN captures user interaction with a video player. The text of the video is automatically semantically annotated and linked to the DBpedia ontology with an entity classifier. GAIN output, the aggregated user data, is subject to preference (rule) learning. Both the entity classifier (`entityclassifier.eu`) and the rule learner (`easyminer.eu`) are sister projects of GAIN (`inbeat.eu`).

*Also affiliated with the Dep. of Information and Knowledge Engineering, University of Economics, Prague

†Also affiliated with the Multimedia and Vision Research Group, Queen Mary, University of London

[1]General Analytics INterceptor and INterest Aggregator

RecSys'13, October 12–16, 2013, Hong Kong, China.
ACM 978-1-4503-2409-0/13/10.
http://dx.doi.org/10.1145/2507157.2508217.

1. SYSTEM OVERVIEW

The input of GAIN is a variable number of *interest clues* and a *semantic description of content*. Interest clues are user interactions, which can be either explicit (clicking a button) or implicit (attention level). Content is described in terms of *views*, the smallest amount of content to which the interest clue can be related to. View can contain multiple *objects* linked to a domain ontology by Linked Open Data (LOD) identifiers. Objects are described by attributes, properties which influence user interest in the object. Their values might be literals, or LOD identifiers. Domain ontology can be loaded to GAIN allowing computation of relatedness between attribute values.

GAIN output is a set of instances in a tabular form suitable for processing with generic machine-learning algorithms. Each mining instance relates to one view and is described by a fixed-length vector of weighted classes from the domain ontology, created by an aggregation of content of objects in the view, and a scalar value of *interest*, aggregation of all preference clues related to the view.

To foster integration with other components of a recommender system, GAIN is encapsulated into a REST web service interface. In the remainder of this section, the internals of GAIN are described.

Tracking module is integrated into the user interface with which the users interact. It is responsible for capturing interest clues and forwarding them in the form of events to the Application module. There are two version: modified Google Analytics (GA) JavaScript tracking code (`ga.js`) for web analytics and a REST API for other domains.

Application module contains the GAIN core logic. GAIN is implemented on the Node.js (`http://nodejs.org/`), a lightweight scalable platform with event-driven, non-blocking I/O model. After processing, the data are passed to the storage module. API for importing and exporting data at multiple processing stages is also available.

Storage module is implemented as a NoSQL document-oriented database MongoDB (`http://www.mongodb.org/`). The key advantage of this solution is a schema-less design, which fosters the attachment of semantic description to objects, with which the user interacted.

Aggregation module combines the interest clues from the Tracking module with description of content interacted with into fixed-length vectors suitable for machine learning. Content annotation is sent to the tracking module simultaneously with each interaction or added during the aggregation.

Figure 1: An extended version of the YouTube player used for the SmartTV demo

2. GAIN IN SMART TV RECOMMENDER

The LinkedTV project (`linkedtv.eu`) develops a recommender system for "Smart TV", which gathers data on televiewers behaviour and uses them to learn their preferences. These are then used to choose links to content on the web related to the video being played that will be recommended to the user. GAIN is used for capturing and preprocessing user feedback during video playback. For easy accessibility, this usecase is demoed with the YouTube player.

User Tracking

There are two sources of interaction: remote control and user behaviour feedback. These two types of interaction are depicted at Fig. 1 as "Basic controls" and "Behaviour controls". Within the LinkedTV project, the latter type of interaction comes from 3D head pose estimation [2]. Every user interaction is immediately sent by the player via the REST API component to GAIN server along with the time in the video and video identification.

Consider example televiewer watching news. During a spot covering a football match, the user clicks on a bookmark button and behaviour tracking records "Looking at screen".

Ontology-based Content Description

The video is fragmented into smaller time-intervals in order to relate user feedback more precisely to what is on the screen at the time the feedback is recorded. This segmentation is performed from start and stop time of subtitles, in the LinkedTV platform, shot detection algorithms are used. The result of segmentation are *views*.

To identify salient objects, the text of the video, as extracted from subtitles, is processed with entity recognition tool THD [1]. The THD tool also assigns each object with several *type attributes* corresponding to one or more DBpedia Ontology classes.

Example: if entity dbpedia:Ronaldinho is recognized in the subtitle, the type assigned is dbpedia-owl:SoccerPlayer.

Aggregation

Multidimensional semantic (taxonomical) description of tracked objects are processed along with user feedback to the lower-dimensional output representation, with a tabular form suitable for analysis with mainstream data mining algorithms.

Content Description. Each *view* contains a varying number of objects. First, the domain ontology is used to expand the *type* attributes to a *class membership vector*.

Next, the class membership vectors of all objects within a view are aggregated into a single class membership vector.

The "dbpedia-owl:SoccerPlayer" type attribute value is expanded to subvector containing non-zero values also for its superclasses: dbpedia-owl:Athlete and dbpedia-owl:Person.

User feedback. None to multiple user actions are recorded during a view. These are aggregated into a single *interest value*, which ranges from -1 (maximum disinteret) to 1 (maximum interest). The impact of individual types of actions is determined by a custom set of rules, such as *if user clicks on bookmark, the interest is increased by 0.5* or *if user looks at the screen, the interest is increased by 0.3. Our user's bookmark and look at screen actions resulted in a total 0.8 interest for the view.*

Learning preference rules with EasyMiner

For a given user, GAIN outputs one table, where rows correspond to views recorded for the user. The Table has $n+m+1$ columns, where n is the number of classes in the domain ontology, m number of unique objects and the last column is the estimated value of interest. User preferences are discovered with EasyMiner [5], a web sevice and web application for association rule discovery based on LISp-Miner [4].

Mining interactions of our user yields rules such as:
$\circ$ *dbpedia-owl:Athlete* $\wedge$ *dbpedia:Brazil* $\rightarrow$ *interest* $= \langle 0.3; 0.5 \rangle$
$\circ$ *dbpedia:Ronaldinho* $\rightarrow$ *interest* $= (0.5; 1\rangle$.

Content recommendation with Business Rules

EasyMiner allows to export selected rules to a Business Rules Engine [3]. There is an ongoing work on using this engine as a recommender system. Input is a description of a candidate content item with an $n+m$ dimensional vector of concepts. The output is the predicted interest.

Acknowledgment. This research was supported by the EU via the LinkedTV project (FP7-287911), CTU in Prague grant (SGS13/100/OHK3/1T/18) and VSE by grant IGA 20/2013. The content in Fig. 1 was adapted from Wikipedia.

3. REFERENCES

[1] M. Dojchinovski and T. Kliegr. Entityclassifier.eu: real-time classification of entities in text with Wikipedia. In *ECML'13*, pages 654–658. Springer.

[2] J. Leroy, F. Rocca, M. Mancas, and B. Gosselin. Second screen interaction: an approach to infer tv watcher's interest using 3d head pose estimation. In *LiME-2013*, WWW '13 Companion, ACM, pages 465–468, 2013.

[3] S. Vojíř, T. Kliegr, A. Hazucha, R. Škrabal, and M. Šimůnek. Transforming association rules to business rules: EasyMiner meets Drools. In *RuleML-2013 Challenge*. CEUR-WS.org, 2013.

[4] M. Šimůnek. Academic KDD project LISp-Miner. In *Advances in Soft Computing - Intelligent Systems Design and Applications*, pages 263–272. Springer, 2003.

[5] R. Škrabal, M. Šimůnek, S. Vojíř, A. Hazucha, T. Marek, D. Chudán, and T. Kliegr. Association rule mining following the web search paradigm. In *ECML'12*, pages 808–811. Springer, 2012.