Information Extraction from HTML Product Catalogues: from Source Code and Images to RDF

Martin Labský, Vojtěch Svátek and Ondřej Šváb
Department of Information and Knowledge Engineering,
University of Economics, Prague, W. Churchill Sq. 4, 130 67 Praha 3, Czech Republic
{labsky,svatek,xsvao06}@vse.cz

Pavel Praks, Michal Krátký and Václav Snášel
Departments of Applied Mathematics and of Computer Science,
VŠB – Technical University of Ostrava, 17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
{pavel.praks,michal.kratky,vaclav.snasel}@vsb.cz

Abstract

We describe an application of information extraction from company websites focusing on product offers. A statistical approach to text analysis is used in conjunction with different ways of image classification. Ontological knowledge is used to group the extracted items into structured objects. The results are stored in an RDF repository and made available for structured search.

1 Introduction

One of hot issues in WWW research is to enable augmentation of the human-consumable web content with machine-consumable *semantic web* (SW) content. Although the very initial concept of SW assumed manual annotation of web pages with RDF statements, tools and techniques for *information extraction* (IE) have recently been recognised as one of key enablers for semantic web scaling.

Among important targets of IE for the SW are HTML *product catalogues*. Since emphasis is put on attractive presentation, structure of catalogues is very diverse and often differs even within a single website. This is why *wrapper-based* approaches to IE [7, 9], which rely on regular page structure, cannot always be applied. We therefore chose a statistical approach to IE that relies on the *content* of the extracted items and on the *context* in which they appear.

In this paper, we focus on a pilot application in the domain of *bicycle product offers*. Section 2 presents automatic HTML annotation based on Hidden Markov Models (HMMs), which is augmented with image analysis in Section 3. Section 4 describes how instances are composed from annotations using an ontology. Sections 5 and 6 outline utilised XML data storage tool, RDF storage of extracted results, and show the search interface. Finally, related and future work are discussed in Sections 7 and 8.

2 Web Page Annotation Using HMMs

HMMs are probabilistic finite state machines which represent text as a sequence of tokens. An HMM consists of *states*, which *generate* tokens, and of *transitions* between these states. States are associated with token generation probabilities, and transitions with transition probabilities. Both kinds of these probabilities are estimated from training data using maximum likelihood. For the purpose of IE, some states are associated with semantic tags to be extracted. To annotate a document using a trained HMM, the document is assumed to have been generated by that HMM. The most probable state (i.e. tag) sequence is then found using the Viterbi algorithm [13].

Tokens modelled by our HMM include *words*, *formatting tags* and *images*. The chosen HMM structure is inspired by [5] and is sketched in Figure 1. Extracted slots are modelled by *target* states (T). Each target state is equipped with two helper states that represent the slot's typical context – the *prefix* and *suffix* states (P and S). Irrelevant tokens are modelled by a single *background* state (B). Contrary to [5], which use separate HMMs for each slot, we train a single large HMM to extract all slots at once. Our model thus contains multiple target, prefix and suffix states. This approach, also used in [1], captures relations between nearby slots (e.g. a product image often follows its name).

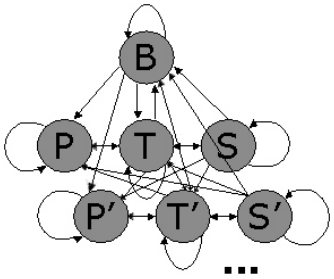

Figure 1. HMM architecture

To train our model, we manually annotated 100 HTML documents with 15 semantic tags, which included bicycle *name*, *price*, *picture*, its *properties* and *components*. The documents belonged to different websites (there were typically only 3–4 documents from the same shop) and thus had heterogenous formattings; the sites were picked from the Google Directory *Sports-Cycling-BikeShops-Europe-UK-England*. Each document contained from 1 to 50 bicycle offers, and each offer at least contained the name and price. There were 1034 offers consisting of 4126 annotations[1]. Similarly to [11], preprocessing amounted to conversion to XHTML, and to rule-based replacement of several frequently used patterns (such as "add to basket" buttons) by dedicated tokens.

3 Impact of Image Classification

As a baseline approach, all images were represented by identical tokens and product pictures could only be distinguished based on the context in which they appeared. Alternatively, we coupled the tagger with image classifiers to provide it with more information. We used the following features for classification: *image size*, *similarity* to training product images, and whether there was *more than one occurrence* of the same image in the containing document.

3.1 Image size

We modelled size of bicycle images using a 2-dimensional normal distribution N, only estimated from a collection of positive training examples C. The dimensions x, y of a new image I were first evaluated using the estimated normal density N. The density value was then normalised to (0,1) using the density's maximum value, N_{max}.

$$Siz_C(I) := \frac{N(x,y)}{N_{max}} \qquad (1)$$

An image I was classified as Pos or Neg by comparing its $Siz_C(I)$ score to a threshold which was estimated by minimising error rate on an additional held-out set of images.

[1]Training data and IE demo are at `http://rainbow.vse.cz`.

Figure 2. Example of image similarity

Within our document collection, image size appeared to be the best single predictor with error rate of 6.6%. However, this was mainly due to our collection being limited to relevant product catalogues only. With more heterogenous data, the actual image content will become necessary.

3.2 Image similarity

We experimented with a *latent semantic* approach to measuring image similarity [12], previously applied to similarity-based retrieval of images from collections. As an example, see the four images in Fig. 2. The bike on image (a) is a training case, with similarity 1 to itself; the similarity is high for another bike (b), lower for a moped (c), and close to zero for a bicycle bag (d). We used this image-to-image similarity measure $Sim(I,J)$ to compute $Sim_C(I)$, the similarity of an image I to a *collection* of images C. In our experiments, C contained the training bicycle pictures (positive examples only). We compute $Sim_C(I)$ using K-nearest neighbour approach by averaging the similarities of the K most similar images from the collection.

$$Sim_C(I) = \frac{\sum_{K\ best\ images\ J \in C} Sim(I,J)}{K} \qquad (2)$$

Experimentally, we set $K = 20$ since lower values of K lead to a decrease in robustness since $Sim_C(I)$ became too sensitive to individual images J, and higher values did not bring further improvement. The similarity-based classifier achieved an error rate of 26.7%, with the decision threshold for $Sim_C(I)$ estimated again on held-out images.

3.3 Combined classifier

For the combined image classifier, we used as features the above described size-based score $Siz_C(I)$, similarity score $Sim_C(I)$ and a binary feature indicating whether the image occurs more than once in its document. Among classifiers available in the *Weka* [15] environment, the best error rate of 4.8% (*without* using held-out data) was achieved

by *multilayer perceptron*. All results were measured using 10-fold cross-validation on a set of $1,507$ occurrences of 999 unique images taken from our training documents. The cross-validation splits were made at document level, i.e. all images from one document were either used for training or for testing. The first two classifiers used additional 150 held-out images to estimate their decision thresholds.

3.4 Using Image Information for Extraction

To improve IE results, we replaced each image occurrence in document with the predicted class of that image. Since binary decisions would leave little room for the HMM tagger to fix incorrect classifications, we adapted the above-described classifiers to classify into *three* classes: Pos, Neg, and Unk. In this way, the HMM tagger learnt to tag the Pos and Neg classes correspondingly, and the tagging of the Unk class depended more strongly on the context. To build ternary size- and similarity-based classifiers, we penalised each wrong decision with a *cost* of 1. The cost of Unk decisions was set experimentally in the range $(0, 1)$ so that the classifier produced 5-10% of Unk decisions on the held-out set. For the combined ternary classifier, we achieved best results with a Weka decision list shown in Table 1. The list combines image occurrence count with predictions of the size- and similarity-based ternary classifiers, denoted as $class^3_{Siz}$ and $class^3_{Sim}$, respectively.

Table 1. Combined ternary classifier

Order	Rule
1	$class(I) = Neg$ if$(occurences(I) > 1)$
2	$class(I) = Pos$ if$(class^3_{Siz}(I) = Pos)$
3	$class(I) = Unk$ if$(class^3_{Siz}(I) = Unk)$
4	$class(I) = Unk$ if$(class^3_{Sim}(I) = Pos)$
5	$class(I) = Neg$

We evaluated IE results with all three ternary classifiers and compared the results to the case where no image information was available. The new image information from the combined classifier lead to an increase of 19.1% points in picture precision and also to subtle improvements for other tags. Improvements in precision, recall and F-measure for 3 frequent slots (product pictures, names and prices), on a per-token basis, are shown in Table 2 for all three classifiers.

4 Ontology-Based Instance Composition

In order to get structured product offers, annotated attributes from previous section need to be grouped into (bicycle offer) instances. We use a simple sequential algorithm that exploits constraints defined in a tiny *presentation ontology* (similar to IE ontologies in [4]) which encodes *optionality* and *cardinality* of attributes. These constraints partly pertain to the domain and partly to the way of presenting information in web catalogues. The algorithm adds an annotated attribute to the currently assembled instance unless it would cause inconsistency; otherwise, the current instance is saved and a new instance created to accommodate this and the following items. Although the algorithm correctly groups about 90% of attributes on hand-annotated data, on noisy automatically annotated data its performance drops to unsatisfactory 50%, often due to single missing or extra annotations. This is a subject of ongoing research.

Table 2. IE results for selected tags

Tag	Prec	Rec	F	Prec	Rec	F
	No image information			Image similarity		
Picture	67.8	87.1	76.2	78.5	87.3	82.7
Name	83.7	82.5	83.1	83.9	82.5	83.2
Price	83.7	94.4	88.8	84.0	94.4	88.9
	Image size			Combined		
Picture	85.6	88.4	87.0	86.9	89.1	88.0
Name	83.8	82.5	83.1	83.8	82.5	83.2
Price	84.0	94.4	88.9	84.0	94.4	88.9

5 Generic System Infrastructure

For *input data* storage, we adopted the XML & full-text indexing and query engine *AmphorA*. It stores each root-to-leaf path in an XML document as a point in multi-dimensional space, where each dimension corresponds to a level in the XML tree [8]. The full power of the XML storage facility is not yet employed in the product catalogue application. At the moment, *AmphorA* downloads chosen websites, performs XHTML conversion and provides the IE tool with XHTML sources. All components are wrapped as *web services*, and called by a simple client application.

6 Result Storage And Retrieval

For compliance with the SW, we use an ontology based on *RDF Schema* to store the extracted instances in RDF. As RDF repository we chose *Sesame*[2] because of its SQL-like declarative query language *SeRQL*, which is used by our online search tool[3]. Its interface, shown in Fig. 3, allows users to search for offers based on attribute values or ranges and to further navigate through the repository.

7 Related Work

The number of existing web IE tools is quite high. Recently reported IE tools for SW are *S-CREAM* [6] and

[2] `http://www.openrdf.org`
[3] `http://rainbow.vse.cz:8000/sesame/`

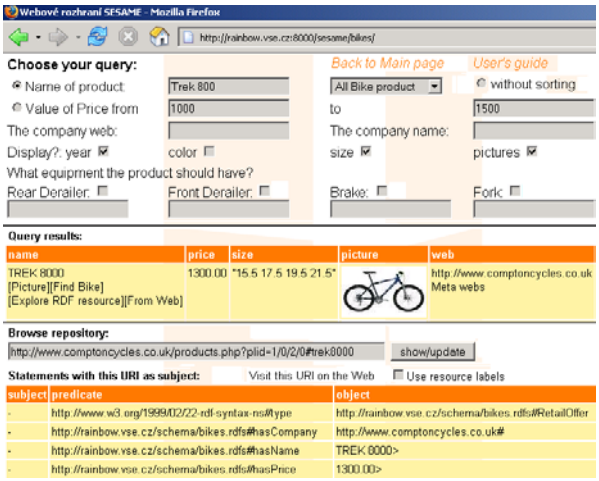

Figure 3. Online search interface

MnM [14]. They attempt to integrate the processes of mark-up of training data and subsequent automated extraction from new data. *Armadillo* [3] and *Pankow* [2], in turn, rely on bootstrapping training data from existing resources, which minimises human annotation effort.

In comparison, our project combines IE from text with *image analysis* and integrates it with subsequent *end-user retrieval* of extracted results. We also focus on *company websites*, which are not frequently targeted by academic IE research; presumably, they exhibit less transparent logical structures and fewer data replications than e.g. computer science department pages or bibliographies. Product websites were addressed by the *CROSSMARC* project [11], it however did not seem to pay particular attention to presentation of extracted results in semantic web format. Its emphasis was on multi-linguality, and hence was more NLP-oriented than our current study.

8 Conclusions and Future Work

We described an application of IE from HTML and images, leading to searchable RDF result repository. The application uses web services to integrate the IE tool with XML storage system. In the *IE* tool, we plan to experiment with more advanced statistical models, such as *Conditional Random Fields* [10], which cope better with mutually dependent textual items. We also need to replace the baseline implementation of ontology-based *instance composition* with a statistical parser that would be robust on automatically annotated data. For some of the layout-based problems mentioned in Section 4, heuristics from [3, 4] can be applied. We also consider using an adapted Viterbi algorithm [1] respecting constraints defined in our presentation ontology. Furthermore, the XML data query facility *AmphorA* will soon support a subset of XPath language, which can be used by IE tool e.g. for efficient querying of multiple documents from the same website. Finally, we plan to *bootstrap* [2] our limited training data using web search engines and data picked from *public resources*.

The research is partially supported by the Czech Science Foundation grant no. 201/03/1318.

References

[1] V. Borkar, K. Deshmukh, and S. Sarawagi. Automatic segmentation of text into structured records. In: SIGMOD 2001.

[2] P. Cimiano and S. Staab. Learning by Googling. In: SIGKDD Explorations 2004.

[3] F. Ciravegna, S. Chapman, A. Dingli, and Y. Wilks. Learning to Harvest Information for the Semantic Web. In: ESWS 2004.

[4] D.W. Embley, C. Tao, and S.W. Liddle. Automatically extracting ontologically specified data from HTML tables with unknown structure. In: Proc. ER 2002.

[5] D. Freitag and A. McCallum. Information extraction with HMMs and shrinkage. In: AAAI Workshop on Machine Learning for IE 1999.

[6] S. Handschuh, S. Staab, and F. Ciravegna. S-CREAM – Semi-automatic CREAtion of Metadata. In: Proc. EKAW 2002.

[7] C.A. Knoblock, S. Minton, J.L. Ambite, N. Ashish, P.J. Modi, I. Muslea, A.G. Philpot, and S.Tejada. Modeling Web Sources for Information Integration. In: Proc. AAAI WI 1998.

[8] M. Krátký, J. Pokorný, and V. Snášel. Implementation of XPath Axes in the Multi-dimensional Approach to Indexing XML Data. In: Proc. Current Trends in Database Technology, DataX, EDBT 2004.

[9] N. Kushmerick, D. S. Weld, and R. Doorenbos. Wrapper Induction for Information Extraction. In: Proc. Intl. Joint Conference on Artificial Intelligence 1997.

[10] J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proc. ICML 2001.

[11] M.T. Pazienza, A. Stellato, and M. Vindigni. Combining ontological knowledge and wrapper induction techniques into an e-retail system. In: ECML/PKDD workshop ATEM 2003.

[12] P. Praks, J. Dvorský, and V. Snášel. Latent semantic indexing for image retrieval systems. In: Proc. SIAM Conf. on Applied Linear Algebra 2003.

[13] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In: Proc. IEEE 1989.

[14] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, and F. Ciravegna. MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup. In: Proc. EKAW 2002.

[15] I.H. Witten and E. Frank: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann 1999, 1-55860-552-5.