Multimedia information extraction from HTML product catalogues

Martin Labský[1], Pavel Praks[2], Vojtěch Svátek[1], Ondřej Šváb[1]

[1] 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
[2] Department of Mathematics and Descriptive Geometry,
Department of Applied Mathematics, Technical University of Ostrava,
17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
pavel.praks@vsb.cz

Abstract. We describe a demo application of information extraction from company websites, focusing on bicycle product offers. A statistical approach (Hidden Markov Models) is used in combination with different ways of image classification, including latent semantic analysis of image collections. 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.

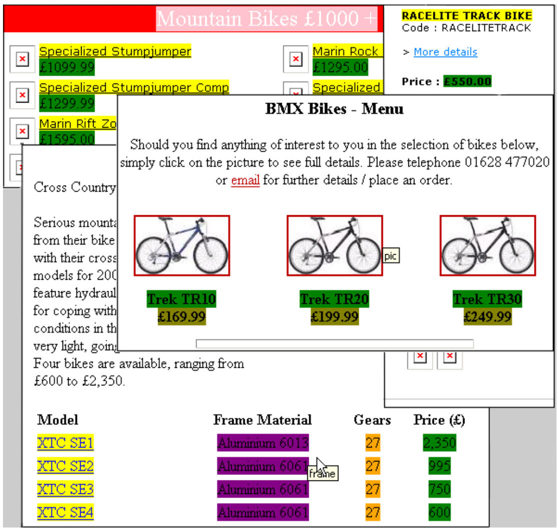
1 Introduction

Tools and techniques for *web information extraction* (WIE) have recently been recognised as one of key enablers for semantic web (SW) scaling. In our long-term project named *Rainbow*[3] we address several intertwined topics that we consider important for efficient 'WIE for SW' applications:

1. Exploitation of *multiple information modalities* available in web documents
2. Synergy of *learning* and reuse of *ontological information*
3. Automated acquisition and labelling of *training data* for extractor learning
4. Bridging between automated acquisition of SW data and their *usage*
5. Support for easy design of WIE applications *from components.*

In this paper, we focus on an ongoing demo application in the domain of *bicycle product offers.* Section 2 presents the core method: automated HTML annotation based on Hidden Markov Models. Section 3 extends the analysis of HTML code with that of images. Section 4 describes the composition of product offer instances with the help of a simple ontology. Section 5 outlines the architecture of the demo application and the subsequent usage of extracted data in an RDF repository. Finally, section 6 focuses on future work.

[3] http://rainbow.vse.cz



Mountain Bikes £1000 +

Specialized Stumpjumper
£1099.99

Specialized Stumpjumper Comp
£1299.99

Marin Rift Zo
£1595.00

Marin Rock
£1295.00

Specialized

RACELITE TRACK BIKE
Code : RACELITETRACK

> More details

Price : £550.00

BMX Bikes - Menu

Should you find anything of interest to you in the selection of bikes below, simply click on the picture to see full details. Please telephone 01628 477020 or email for further details / place an order.

Trek TR10
£169.99

Trek TR20
£199.99

Trek TR30
£249.99

Cross Country

Serious mounta
from their bike
with their cross
models for 200
feature hydraul
for coping with
conditions in th
very light, going
Four bikes are available, ranging from
£600 to £2,350.

Model	Frame Material	Gears	Price (£)
XTC SE1	Aluminium 6013	27	2,350
XTC SE2	Aluminium 6061	27	995
XTC SE3	Aluminium 6061	27	750
XTC SE4	Aluminium 6061	27	600

Fig. 1. Hand-annotated training data

2 Web Page Annotation Using HMMs

For extracting product entries from web catalogues, we built a Hidden Markov Model (HMM) tagger, which assigns a semantic tag to each token from a document. Tokens are either words, formatting tags or images. In our experiments, we evaluated the HMM performance on a diverse set of web pages, which come from different web sites and have heterogenous formattings.

We manually annotated a set of 100 HTML documents chosen from the Google Directory *Sports-Cycling-BikeShops-Europe-UK-England*. Each document contains from 1 to 50 bicycle offers, and each offer consists of at least the bicycle name and price. There are typically 3–4 documents from the same shop in the data. Annotations for 15 bicycle characteristics were made using SGML tags[4]. A sample annotated data is shown in Figure 1.

To represent web documents, we employed extensive pre-processing. Similarly to [7], we transform each document into XHTML and perform canonicalisation of XML entities[5]. Certain HTML tags and tag groups are replaced by their generalisations[6]. Since only words and images can be extracted, we dispose of mark-up blocks that do not directly contain words or images.

HMMs are probabilistic finite state machines, which represent text as a sequence of tokens. An HMM consists of *states* which generate tokens, and of

[4] The training data and a demo are available at `http://rainbow.vse.cz`.

[5] This step unifies different ways of writing the same characters in XML.

[6] Most tags are only represented using their names, disregarding any attributes. Often-occuring design patterns, such as add-to-basket buttons, are identified using several manually authored rules, and replaced by dedicated tokens.

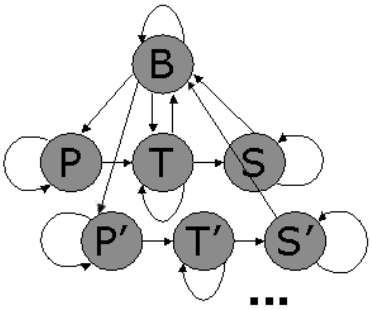


Fig. 2. HMM architecture

transitions between these states. States are associated with token generation probabilities, and transitions with transition probabilities. Both kinds of these probabababilities are estimated from training data. For the purposes of information extraction, states are typically associated with semantic tags to be extracted. To annotate a document using a trained HMM, that document is assumed to have been generated by that HMM. The most probable state sequence is then found using the Viterbi algorithm [12].

The structure of our HMM is inspired by [6] and is sketched in Figure 2. Extracted slots are modelled using *target* states (denoted as T). Each target state is accompanied by two types of helper states responsible for representing the slot's characteristic context – the *prefix* and *suffix* states (P and S). Irrelevant tokens are modelled by a single *background* state (B). Contrary to [6] and [17], which use independent HMMs trained for each slot separately, we train a single composite HMM capable of extracting all slots at once. Our model thus contains multiple target, prefix and suffix states. This approach, also used in [1], captures the ordering relations between nearby slots (e.g. product image often follows its name). We experimented also with other HMM architectures, with results presented in [16].

3 Impact of Image Classification

For the purpose of extracting product images, we examined the impact of image information available to the HMM tagger. As a baseline approach, we measured the tagging performance when no image information was available for tagging. In this case, all images were represented by the same token and product pictures could only be distinguished based on the context in which they appeared.

In order to provide our tagger with more information, we built image classifiers to determine whether the extracted product is depicted in a particular image. We used the following features for classification: *image dimensions*, *similarity* to training product images, and whether there is *more than one occurrence* of the same image in the containing document.

3.1 Image dimensions

For our domain, we modelled images of bicycles using a 2-dimensional normal distribution, only estimated from positive training examples[7]. The dimensions x, y of a new image I are first evaluated using the estimated normal density N. The density value is then normalized to the interval (0,1) using the density's maximum value N_{max}.

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

An image I is then classified as Pos or Neg by comparing its $Dim(I)$ score to a threshold T_{Dim}. This threshold was estimated by minimizing the classification error rate on a separate heldout set of 150 images.

$$class(I) = \left\{ \begin{array}{l} Pos\ if(Dim(I) \geq T_{Dim}), \\ Neg\ otherwise. \end{array} \right. \tag{2}$$

Within our document collection, image dimensions appeared to be the best single predictor with the error rate of 6.6%. However, this is mainly due to our collection being limited to relevant product catalogues only. When dealing with more heterogeneous data, features describing the actual image content will become necessary.

3.2 Image similarity

We experimented with a *latent semantic* approach to measuring image similarity, described in [10] and [11]. This kind of image similarity has been applied to image retrieval from collections, where the task often is to find the most similar image to a query. 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). To compute $sim_C(I)$, we used the K nearest neighbor approach and averaged the similarities of the K most similar images from the collection.

$$sim_C(I) = \frac{\sum_{K\ best\ images\ J \in C} sim(I, J)}{K} \tag{3}$$

Experimentaly, we set $K = 20$, since lower values of K lead to a decrease in the similarity's robustness[8] and higher values did not bring further improvement. To build a classifier, a similarity threshold T_{Sim} was estimated on a heldout set in the same way as for the dimension classifier above. The error rate of the classifier was 26.7% on our document collection.

[7] The positive examples comprise of *all* bicycle pictures found in the documents, not only those labeled as parts of bicycle offers. For information extraction, this increases the role of image context for correct tagging.

[8] With low values of K, $sim_C(I)$ became too sensitive to individual images J with misleading values of $sim(I, J)$.

3.3 Combined classifier

For the combined image classifier, we used the above described dimension score $Dim(I)$, similarity score $Sim(I)$ and a binary feature indicating whether the image occurs more than once in the document. We experimented with different classifiers available in the Weka[9] environment, and the best error rate[10] of 4.8% was achieved by the *multilayer perceptron algorithm.*

Results for all three classifiers are compared in Table 1. All results were measured using 10-fold cross-validation on a set of $1,507$ occurences of 999 unique images taken from our training documents. The first two algorithms used additional 150 heldout images to estimate their decision thresholds. The cross-validation splitting was done at the level of documents, so that all images from a single document were either used for training or for testing.

Table 1. Image classification results

	Dimension	Similarity	Combined
Error rate (%)	6.6	26.7	4.8

3.4 Using Image Information for Extraction

To improve extraction results, we need to communicate the image classifier's results to the HMM tagger. Currently we do this simply by substituting each image occurence in a document by its class. Since these binary decisions would leave little room for the HMM tagger to fix incorrect classifications, we adapted the above binary classifiers to classify into 3 classes: Pos, Neg, and Unk. In this way, the HMM tagger learns to classify the Pos and Neg classes correspondingly, and the tagging of the Unk class depends more strongly on the context.

To build the ternary versions of the dimension- and similarity-based classifiers, we introduced *costs* for the classifier's decisions. Each wrong decision was penalized by $C_{Miss} = 1$ and the cost of each Unk decision was $C_{Unk} \in (0, 1)$. We set C_{Unk} manually such that the classifier produced 5-10% of Unk decisions on the heldout set. While minimizing the sum of these costs on the heldout set, two thresholds were estimated for both the dimension- and similarity-based classifiers, delimiting their Neg, Unk and Pos decisions.

For the combined ternary classifier, we achieved the best results with a decision list shown in Table 2. The list combines image occurence count with the results of the dimension- and similarity-based ternary classifiers, denoted as $class^3_{Dim}$ and $class^3_{Sim}$ respectively.

We evaluated information extraction results with all three ternary classifiers and compared the results to the case where no image information was available.

[9] `http://www.cs.waikato.ac.nz/~ml`

[10] This error rate comes from 10-fold cross-validation *without* using heldout data.

Table 2. Decision list for the combined ternary classifier

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

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 and recall for 3 chosen slots (product pictures, names and prices), measured on a per-token basis, are shown in Table 3 for all three classifiers.

Table 3. 10-fold cross-validation results for selected tags over 100 documents

Tag	Precision	Recall	F-measure	Precision	Recall	F-measure
	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 Dimensions			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

4 Ontology-Based Instance Composition

Semantic web is not about isolated tagged items but about complex and interrelated entities; we thus need to group the labels produced by automated annotation into instances. We currently use a simple sequential algorithm that exploits constraints defined in a tiny *presentation ontology*[11] [9], which partly pertain to the generic domain (bike offers) and partly to the way of presenting information in web catalogues. Figure 3 shows an experimental presentation ontology containing the class 'Bike offer'. The utilized constraints are *uniqueness*, *multiplicity* and *optionality* of certain properties, the latter two indicated with the `*` and `?` symbols, respectively[12]. In addition, *'sticky' properties* (indicated with `!`) are distinguished: as soon as the value of sticky property is discovered

[11] Similar to 'extraction ontologies' used by Embley [5].

[12] Although not shown in the example, we can also use e.g. property value types or regular expressions.

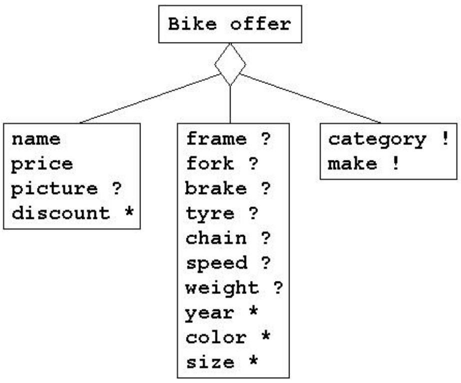


Fig. 3. Bicycle offer presentation ontology

on a page, it is filled to all objects extracted afterwards, until a new value is discovered for this property.

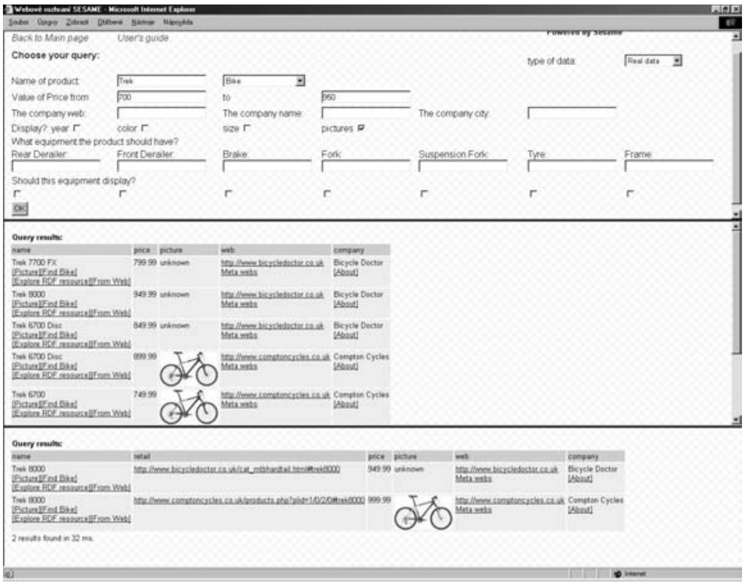
An annotated item is added to the currently assembled (bike offer) instance unless it would cause inconsistency; otherwise, the current instance is saved and a new instance created to accommodate this item and the following ones. Despite acceptable performance on error-free, hand-annotated training data, where the algorithm correctly groups about 90% of names and prices, this 'baseline' approach achieves very poor results on automatically annotated data: on average, less than 50% of corresponding annotations are grouped properly, often for trivial reasons. The most critical problems are connected with *missing or extra annotations*, *multiple different references* to a single slot, and with *transposed HTML tables.*

5 Result Transformation, Storage And Retrieval

All components developed within the *Rainbow* project are wrapped as web services. The WIE component itself is currently being called by a simple *control routine* (written in Java), which also optionally calls other analysis tools: in the bicycle application, we so far experimented with URL-based navigation over the website, extraction of the content of selected META tags, and extraction of 'company profile sentences' from free text[13]. The results are transformed to RDF (with respect to a 'bicycle-offer RDFS ontology') and stored in a *Sesame* [2] repository. An end-user search interface to this repository[14] is shown in Fig. 4. It relies on a collection of *query templates* expressed in SeRQL (the native query language of *Sesame*) and enables a simple form of navigational retrieval [16].

[13] These three approaches to website analysis, implemented independent of the bicycle demo application, are evaluated in [13].

[14] Available at `http://rainbow.vse.cz:8000/sesame`.


Webové rozhraní SESAME - Microsoft Internet Explorer
Soubor Úpravy Zobrazit Oblíbené Nástroje Nápověda
Back to Main page
User's guide
Powered by Sesame
OK
Internet

Choose your query:

type of data: Real data

Name of product: Trek Bike

Value of Price from: 700 to 950

The company web: The company name: The company city:

Display?: year □ color □ size □ pictures ☑

What equipment the product should have?

Rear Derailer: Front Derailer: Brake: Fork: Suspension Fork: Tyre: Frame:

Should this equipment display?

□ □ □ □ □ □ □

Query results:

name	price	picture	web	company
Trek 7700 FX [Picture][Find Bike] [Explore RDF resource][From Web]	799.99	unknown	http://www.bicycledoctor.co.uk Meta webs	Bicycle Doctor [About]
Trek 8000 [Picture][Find Bike] [Explore RDF resource][From Web]	949.99	unknown	http://www.bicycledoctor.co.uk Meta webs	Bicycle Doctor [About]
Trek 6700 Disc [Picture][Find Bike] [Explore RDF resource][From Web]	849.99	unknown	http://www.bicycledoctor.co.uk Meta webs	Bicycle Doctor [About]
Trek 6700 Disc [Picture][Find Bike] [Explore RDF resource][From Web]	899.99		http://www.comptoncycles.co.uk Meta webs	Compton Cycles [About]
Trek 6700 [Picture][Find Bike] [Explore RDF resource][From Web]	749.99		http://www.comptoncycles.co.uk Meta webs	Compton Cycles [About]

Query results:

name	retail	price	picture	web	company
Trek 8000 [Picture][Find Bike] [Explore RDF resource][From Web]	http://www.bicycledoctor.co.uk/cat_mtbhardtail.html#trek8000	949.99	unknown	http://www.bicycledoctor.co.uk Meta webs	Bicycle Doctor [About]
Trek 8000 [Picture][Find Bike] [Explore RDF resource][From Web]	http://www.comptoncycles.co.uk/products.php?plid=1/0/2/0#trek8000	999.99		http://www.comptoncycles.co.uk Meta webs	Compton Cycles [About]

2 results found in 32 ms.

Fig. 4. End-user search interface

6 Future Work

Most urgently, we need to replace the 'toy' implementation of ontology-based *instance composition* with a version reasonably robust on automatically annotated data. For some of the *layout-oriented* problems mentioned in section 4, partial solutions recently suggested in IE research (e.g. [3, 5]) could be reused. We also consider introducing HMMs even to this phase of extraction; a modified version of Viterbi algorithm supporting domain constraints (such as those in our presentation ontology) has already been described in [1]. Another aspect worth investigation is the possibility of (semi-)automatic construction of presentation ontologies from the corresponding *domain ontologies.*

A critical bottleneck of ML-based IE methods (in particular of statistical ones) is the volume of *labelled training data* required. In our experiments with product catalogues, we noticed that the tagger often classifies most product entries correctly but misses a few product names that are very different from the training data. We developed a simple symbolic algorithm that identifies similar *structural patterns* in a document. For example, the HTML tag sequence `<td> <a> <font> <br/> </font> </a> </td>` with arbitrary words in between appears 34 times in one of our training documents: the tagger successfully annotated 28 product names contained in these patterns between `<font>` and `<br/>`, but missed the remaining 6. In such cases, we could collect the remaining product names and use them to enrich the model's training data. By learning novel product names from these 'easy' pages, the model will learn to also recognise them in less structured documents[15]. We also plan to bootstrap the method with data picked from *public resources* related to product offering, following up with our earlier experiments with Open Directory headings and references [8].

Another important task is to replace hard-coded *control routines* with semi-automatically constructed, implementation-independent application models. A knowledge modelling framework has already been introduced for this purpose [14]; currently we examine the adaptability of a PSM-based semantic *web-service configuration* technique in connection with this framework [15].

Eventually, we plan to associate our efforts with the popular *Armadillo* project [3], with which we share most of our abovementioned research interests.

The research is partially supported by grant no.201/03/1318 of the Grant Agency of the Czech Republic, "Intelligent analysis of the WWW content and structure".

References

1. Borkar V., Deshmukh K., Sarawagi S.: Automatic segmentation of text into structured records. In: SIGMOD Conference, 2001.
2. Broekstra J., Kampman A., van Harmelen F.: Sesame: An Architecture for Storing and Querying RDF and RDF Schema. In: Proc. ISWC 2002, Springer LNCS no. 2342.
3. Ciravegna, F., Chapman, S., Dingli, A., Wilks, Y.: Learning to Harvest Information for the Semantic Web. In: ESWS-04, Heraklion, Springer LNCS 2004.
4. Dingli A., Ciravegna F., Guthrie D., Wilks Y.: Mining Web Sites Using Unsupervised Adaptive Information Extraction. In: EACL, 2003.
5. Embley, D.W., Tao, C., Liddle, S.W.: Automatically extracting ontologically specified data from HTML tables with unknown structure. In: ER2002, Tampere 2002, 322-337.
6. Freitag D., McCallum A.: Information extraction with HMMs and shrinkage. In: Proceedings of the AAAI-99 Workshop on Machine Learning for IE, 1999.
7. Grover C., McDonald S., Gearailt D., Karkaletsisy V., Farmakiotouy D., Samaritakisy G., Petasis G., Pazienza M., Vindigni M., Vichotz F., Wolinskiz F.: Multilingual XML-Based Named Entity Recognition for E-Retail Domains. In: LREC Conference, Las Palmas, 2002.
8. Kavalec, M., Svátek, V.: Information Extraction and Ontology Learning Guided by Web Directory. In: ECAI Workshop on NLP and ML for ontology engineering. Lyon 2002.
9. Labský, M., Svátek, V., Šváb, O.: Types and Roles of Ontologies in Web Information Extraction. In: ECML/PKDD04 Workshop on Knowledge Discovery and Ontologies, Pisa 2004.
10. Praks P., Dvorský J., Snášel V.: Latent semantic indexing for image retrieval systems. In: Proceedings of the SIAM Conference on Applied Linear Algebra (LA03), Williamsburg, USA, The College of William and Mary, 2003.
11. Praks P., Machala L., Snášel V.: Iris Recognition Using the SVD-Free Latent Semantic Indexing. In: MDM/KDD 2004 - Fifth International Workshop on Multimedia Data Mining, Seattle, USA, 2004.

[15] Similar bootstrapping strategies are shown in [4].

12. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. In: Proceedings of the IEEE, 77(2), 1989.
13. Svátek V., Berka, P., Kavalec, M., Kosek, J., Vávra, V.: Discovering Company Descriptions on the Web by Multiway Analysis. In: Intelligent Information Processing and Web Mining, IIPWM'03., Springer Verlag, 2003.
14. Svátek, V., Labský, M., Vacura, M.: Knowledge Modelling for Deductive Web Mining. In: Proc. EKAW 2004, Springer Verlag, LNCS, 2004.
15. Svátek, V., ten Teije, A., Vacura, M.: Web Service Composition for Deductive Web Mining: A Knowledge Modelling Approach. In: Proc. Znalosti 2005, VSB-TU Ostrava, to appear 2005.
16. Šváb, O., Labský, M., Svátek, V.: RDF-Based Retrieval of Information Extracted from Web Product Catalogues. In: SIGIR'04 Semantic Web Workshop, Sheffield.
17. Valarakos A., Sigletos G., Karkaletsis V., Paliouras G.: A Methodology for Semantically Annotating a Corpus Using a Domain Ontology and Machine Learning. In: RANLP Conference, Borovets, 2003.