Using a Trained Text Classifier to Extract Information

Tina Eliassi-Rad and Jude W. Shavlik
Computer Sciences Department
University of Wisconsin
1210 W. Dayton Street
Madison, WI 53706 USA
+1 608 265 4892
{eliassi, shavlik}@cs.wisc.edu

ABSTRACT
We address the task of extracting information from Web pages containing free-form English. Our approach assumes the presence of a trained page classifier that uses *variable binding* and outputs some sort of numeric confidence in its predictions. We evaluate the use of such a trained classifier to extract information from previously unseen Web pages. Specifically, we use a *generate-and-test* approach, where we create a large set of possible bindings for the information we wish to extract. We then provide each of these possible binding lists in turn, and see which settings produce the most confident output by the trained classifier. Our particular approach uses neural networks. We provide experimental results on a "home-page finder" agent which is able to extract the first name, the middle name (if available), and the last name from the person's home page with 95% accuracy. We compare our approach to some straightforward algorithms as experimental controls, the best of which is only 58% accurate.

Keywords
intelligent Web agents, information extraction, information retrieval, text classification, machine learning, neural networks

INTRODUCTION
The popularity of the World Wide Web has created a surge of interest in tools that are able to classify text and extract information from on-line documents. In this paper, by *text classification*, we mean the process of labeling a page as relevant or irrelevant with respect to a user's query. *Information extraction* is the process of pulling desired pieces of information out of a Web page, such as the price of a product or the author of an article.

Intuitively, information retrieval and information extraction are inverse problems of each other. In information retrieval, you are given a set of keywords and are asked to return a set of relevant documents. In information extraction, you are given a set of documents and are asked to fill in a set of slots in a given template or database schema.

We describe a system that is capable of building text classifiers and then using them to extract information. We call our system WAWA (short for *Wisconsin Adaptive Web Assistant*). WAWA has two subsystems: (*i*) a text-classification system, and (*ii*) a text-extraction system. The text-classification system has been described previously ([10, 11]) and we will only briefly review it here.

Building text classifiers is straightforward in WAWA. The user provides a set of instructions in the form of IF-THEN statements to the system. These instructions describe the desired behavior of the text classifier upon encountering pages and links on the Web. These initial instructions are then "compiled" into two neural networks. The first neural network is called *ScoreThisPage* and it rates the goodness of a page. The second neural network is *ScoreThisLink* and it rates the goodness of a given hyperlink on a page.

ScoreThisPage is a supervised learner. It learns by being trained on user-provided instructions and user-labeled pages. *ScoreThisLink* is a reinforcement learner. It learns by being trained on user-provided instructions and a set of internally created (i.e., without user intervention) training examples. In WAWA, the user-provided instructions are referred to as *advice*. This name emphasizes the fact that an agent does not blindly follow its user's instructions, but instead refines them based on its own experiences and any pages the user chooses to rate. Conversely, the use of user-provided advice greatly reduces the amount of supervised or reinforcement learning that the system needs to undergo in order to reach a desired level of competence.

The WAWA Advice Language
The features WAWA extracts from a Web page (see [10, 11]) constitute the basic constructs of its advice language. These primitives can be combined to create more complex advice statements using logical and numeric constructs.

Of particular relevance to this article is the fact that WAWA's advice language contains *variables*. To understand how variables are used in WAWA, assume that we wish to use the system to create a *home-page finder*. We might wish to give such a system some (very good) advice like:

> If you see the phrase "*?FirstName ?LastName* 's Home Page" in a page's title, then that page is very likely to be what we are seeking.

The leading question marks ('?') indicate variables that are bound upon receiving a request to find a specific person's home page. The use of variables allows the same advice to be applied to the task of finding the home pages of any number of different people.

Our extraction process, which uses WAWA's trained *ScoreThisPage* network, centers on these advice variables. Simply put, given a page from which we wish to extract information, we generate a large number of candidate bindings, and then in turn we provide each set of bindings to the trained network. Neural networks produce numeric outputs, which can be interpreted as probabilities, and our extraction process returns the highest-scoring bindings.

In this paper's experimental study, we used three variables to learn about first, middle, and last names: ?*FirstName*, ?*MiddleName*, and ?*LastName*.

INFORMATION EXTRACTION ALGORITHM

Imagine we have provided some advice to our home-page-finding neural network, refined the advice with some additional supervised learning (via user-labeled pages), and now wish to use the trained network to extract information from a given new page. We can extract first, middle, and last names from new pages by executing the algorithm presented in Table 1.

Table 1: A Simple Information Extraction Algorithm

1. Retrieve the page's source text and tag it with Brill's part-of-speech tagger [3].
2. If several *proper nouns* are adjacent to other, list each word separately and list all possible pairs and triples of these words (preserving the left-to-right order of the words). Discard all duplicate entries.

   The extracted list of single, double, and triple word phrases ("*candidates*") contain the potential instantiations for the target first name and the target last name (and the target middle name for triples).
3. For each single word in the list of candidates, bind that word to the variable ?*LastName*, and find the score that the trained home-page finder produces.
4. For each paired entry on the list, assign the first word to ?*FirstName* and the second word to ?*LastName.* Find the score the home-page finder generates for each instantiation.
5. For each triplet entry on the list, assign the first word to ?*FirstName*, the second word to ?*MiddleName*, and the third word to ?*LastName.* Find the score that the home-page finder generates for each instantiation.

   If the highest score found in Steps 3-5 is above a pre-defined threshold[1], then the new page is considered a home page and the instantiations for one or more of ?*FirstName*, ?*MiddleName*, and ?*LastName* are returned as the extracted first, middle, and last names for that page.

EXPERIMENTS

Originally, we [10] chose the task of building a home-page finder to evaluate WAWA because of an existing system named *Ahoy!* [9]. Ahoy! uses a technique called Dynamic Reference Sifting (DRS), which filters the output of several Web indices and generates new guesses for URLs when no promising candidates are found.

We were able to build our home-page finder in a couple of days. We wrote about 80 general advice rules describing the desired behavior of a home-page finder. To test our agent, we randomly selected 215 people from Aha's list of machine learning and case-based reasoning researchers (www.aic.nrl.navy.mil/~aha/people.html). To reduce the computational load of our experiments, we limited the list to people in the United States. Out of the 215 people selected, we randomly picked 115 of them to train WAWA and used the remaining 100 as our test set.

In our previous work [10], we reported that the WAWA home-page finder was more accurate than several other home-page finders.

It is important to note that we intentionally did *not* provide any advice that is specific to ML, CBR, AI research, etc. It is also important to note that we create these advice rules for the classification task, and had no intention at that time to use them for information extraction.

To test the information-extraction ability of WAWA, we then ran our information-extraction algorithm on the test-set from Aha's ML/CBR list.

We compared our results with several simple name extraction algorithms, in order to better judge our approach's effectiveness. *Algorithm A1* performs Steps 1 and 2 in our information extractor, but simply scores each candidate by the number of times it appears on the Web page. *Algorithm A2* is a slight variant of *A1*, in that it does not add candidates that have three words to the list of potential extractions; in *A2*, a candidate can contain one or two words. *Algorithm A3* is similar to *A2*, but only considers single words.

[1] In this paper's experiments, we set this threshold to zero.

We also explored several variants of algorithm *A1*. Each ranked the candidates by the following equation:

$$\begin{aligned}\text{Score} = \; & \alpha(\#\text{ of "}PN_f\, PN_l\text{" occurrences}) + \\ & \beta(\#\text{ of "}PN_f\, PN_m\, PN_l\text{"}) + \\ & \delta(\#\text{ of } PN_f) + \gamma(\#\text{ of } PN_l)\end{aligned}$$

where the first summand refers to the occurrences of a two-word phrase on the Web page, the second counts the number of the desired three-word phrase (assuming the candidate includes a middle name), and the final two simply count the number of (possibly isolated) occurrences of the candidate's first and last names. The best setting of the weights we found was $\alpha=10$, $\beta=7$, $\delta=1$, and $\gamma=3$. We call this *Algorithm A4*. The other settings we tried produced accuracies within one percentage point of *A4*'s.

Table 2 presents the correctness of the various algorithms on our 100 test pages. An extraction is considered correct when the highest scoring candidate satisfies *all* of the following: (a) the person's last name is bound to *?LastName, (b) ?FirstName* is bound to the person's first name or first initial, or is unbound, (c) *?MiddleName is* bound to the person's middle name or first initial, or is unbound. When several candidates tie for the highest score, the extraction is considered correct if *any* is correct, which favors the *A* algorithms, since WAWA produces continuous outputs, which the others are based on discrete counts.

Notice, for example, that if the highest output produced by the network only binds the first name on the home page, then that extraction is considered incorrect. For example, if the name on the home page is "John Eric Smith", then we consider an extraction to be correct if the highest output produced by the network is associated with one of the following bindings: "Smith," "John Smith," "J. Smith," "John Eric Smith," "John E. Smith," and "J. E. Smith."

Table 2: Results on the Information Extraction Task

System	Testset Accuracy
WAWA's Info. Extractor	95%
Algorithm *A4*	58%
Algorithm *A1*	58%
Algorithm *A2*	58%
Algorithm *A3*	47%

As can clearly be seen, using the system trained to classify Web pages produces a very accurate extractor as well. The correct instantiation for the five test-set examples that our algorithm was not able to extract correctly always appeared in the top-seven highest-ranking instantiations. These errors mistook the department name, the school name, or the research lab's name on the home page for the person's name. WAWA's erroneous extractions were (i) *Computer Science*, (ii*) Louisiana Tech*, (iii) *Harvard Law School*, (iv) *USCS Baskin School*, and (v) *Naval Research Laboratory*. It is interesting to note that none of the errors were due to mistaking another person's name appearing on the page for that of the home page's owner. Hence, it is likely we could further increase accuracy by using such additional information as lists of common names to give *a priori* probabilities to each of the candidate extractions.

RELATED WORK

Only a few researchers have explored the connection between information retrieval and information extraction.

Craven and Kumlien [4] use a sentence classifier to extract instances of a binary relation from text databases in molecular biology. The naive Bayes algorithm with a bag-of-words representation [5] is used to classify sentences. A sentence is classified as a positive example if it contains at least one instance of the target relation.

WAWA's information-extraction system is similar to [4] in that we use a text classifier to extract information. However, in [4], the text classifier classifies small chunks of text (such as sentences) and extracts only the words that are in the given semantic lexicons. In WAWA, the text classifier is able to classify a text document as a whole and generates a lot of extraction candidates without the need for semantic lexicons. Another difference between WAWA and the system described in [4] is that their text classifier is built for the sole purpose of extracting information. However, WAWA was initially built to classify text documents (of any size) and its extraction ability is a side effect of the way the classifier was implemented.

Zaragoza and Gallinari [12] use hierarchical information-retrieval methods to reduce the amount of data given to their stochastic information-extraction system. The standard IR technique of *TF/IDF* weighting [8] is used to eliminate irrelevant documents. Then, the same IR process is used to eliminate irrelevant paragraphs from relevant documents. Relevant paragraphs have at least one extraction template associated with them. Finally, Hidden Markov Models are used to extract patterns at the word level from a set of relevant paragraphs.

WAWA's extractor is different than the system in [12] in that we do not use a text classifier to filter out data and then use another model to extract data. WAWA's text classifier is used directly to extract information.

Riloff and Lehnert [6, 7] describe methods for using the patterns generated by an information-extraction system to classify text. During training, a *signature* is produced by pairing each extraction pattern with the words in the training set that satisfy that pattern. A signature is then labeled as a *relevancy signature* if it is highly correlated with the relevant documents in the training set. In the testing phase, only if a document contains one of the generated relevancy signatures is it labeled as relevant.

Using information-extraction patterns to classify text [6, 7] is a topic of future work for us. There is a big difference between the advice rules given by the user to WAWA and the extraction patterns generated by the information-extraction systems used in [6, 7]. The advice rules define the behavior of the agent upon encountering Web pages and documents. The extraction patterns generated in [6, 7] are learned from a set of labeled training examples and are used to extract words and phrases from sentences.

We have come across two learning systems that focus on extracting names. *IdentiFinder*™ [2] uses a Hidden Markov Model and textual information (such as capitalization and punctuation) to learn to recognize and classify names, dates, times, and numerical quantities. A name is classified into three categories: the name of a person, the name of a location, and the name of an organization. Numerical quantities are classified into monetary amounts or percentages. Baluja et al [1] use a decision-tree classifier in conjunction with information from part-of-speech tagging, dictionary lookup, and textual information (such as capitalization) to extract names. Their system does not attempt to distinguish between names of persons, locations, and organizations.

In our information-extraction system, we do not (yet) use dictionary lookups as evidence. In our next experiments, we will attempt to extract location names and organization names. One advantage of our system compared to IdentiFinder™ and Baluja's system is that we do not need a large number of labeled training examples to achieve high performance. This is because we solve the problem by using theory-refinement techniques, which allow users to easily provide task-specific information via approximately correct inference rules (e.g., our advice rules).

CURRENT AND FUTURE WORK

We have not yet used any advice that references the part-of-speech tags in our experiments. In our future experiments, we will try to take advantage of the tagging information during the classification process. We also plan to extend our approach to use additional sources of information, such as lists of common names, etc. Moreover, we are planning to extend our current extraction algorithm to be able to perform the general named-entity extraction problem. Finally, we are also investigating ways to more tightly couple the processes of information extraction and information retrieval.

CONCLUSION

There is a strong connection between information retrieval and extraction, one that has not been explored fully. We described and evaluated a method for using a trained information-retrieval agent to also perform information extraction. Our agent is built on top of our WAWA system [10, 11], which accepts advice containing variables that are bound at "query time." Our extraction process generates a large set of candidate variable bindings, and uses the trained network to judge which is "best." The highest-scoring binding is returned as the extracted information.

In our reported experiments, we achieved an accuracy of 95% on previously unseen Web pages on the task of extracting the first, the middle (if available), and the last names on a person's home page, an accuracy that greatly exceeded that of our experimental controls.

Our suggested, new approach for information extraction is essentially as follows. Using a system that involves variable binding and produces numeric outputs, first create an automated classifier of Web pages (e.g., *"How likely do you think this is John Smith's home page").* Next, create a set of candidate bindings for the information to be extracted. Finally, use the polished (by hand, via machine learning, or through some combination) classifier to rank the candidates. One potential weakness of our approach is that the number of candidates generated has to be "reasonable," but with today's workstations a large set of candidates can be quickly scored, and our *generate-and-test* approach is likely to be widely applicable.

REFERENCES

1. Baluja, S., Mittal, V., & Sukthankar, R. Applying machine learning for high performance named-entity extraction. *Proc. Pacific Association for Computational Linguistics.* 1999.
2. Bikel, D., Schwartz, R., & Weischedel, R. An algorithm that learns what's in a name. *Machine Learning* 34, 1999.
3. Brill, E., Some advances in rule-based part of speech tagging, *Proc. AAAI '94*, pp. 722-727.
4. Craven, M., & Kumlien, J. Constructing biological knowledge bases by extracting information from text sources, *Proc. 7th Intl. Conf. on Intelligent. Systems for Mol. Biol.*, 1999.
5. Mitchell, T. *Machine Learning*, McGraw-Hill, 1997.
6. Riloff, E. Using learned extraction patterns for text classification. In *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*. Wermter, S., Riloff, E., & Scheler, G. (eds.). pp. 275-289, Springer-Verlag, 1996.
7. Riloff, E., & Lehnert, W. Information extraction as a basis for high-precision text classification. *ACM Transactions on Information Systems* 12:296-333, 1994.
8. Salton, G., & Buckley C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management* 24:513-523, 1988.
9. Shakes, J., Langheinrich, M., & Etzioni, O. Dynamic Reference Sifting: A Case Study in the Homepage Domain, in *Proc. 6th Intl. World Wide Web Conf.*, pp. 189-200, 1997.
10. Shavlik, J., Calcari, S., Eliassi-Rad, T., & Solock, J. An instructable, adaptive interface for discovering and monitoring information on the World-Wide Web, *Proc. IUI '99*.
11. Shavlik, J., & Eliassi-Rad, T. Intelligent agents for Web-based tasks: An advice-taking approach, *AAAI/ICML-98 Workshop on Learning for Text Categorization*.
12. Zaragoza, H., & Gallinari, P. An automatic surface information extraction system using hierarchical IR and stochastic IE. *Proc. ECML '98 Workshop on Text Mining*.