Next Generation Annotation Interfaces for Adaptive Information Extraction

Alexiei Dingli
Department of Computer Science,
University of Sheffield,
Regent Court,
211 Portobello Street,
S1 4DP Sheffield
alexiei@dcs.shef.ac.uk

Abstract

The evolution of the Internet into the largest existent digital library is bringing about new challenges. One of the biggest problems is the location of information. The most promising approach seems to be performing searches semantically however this cannot work without semantically annotated documents. These documents are few and the manual annotation process to make them is both time consuming and error prone. To solve this problem Information Extraction (IE) technologies can be used to automatically annotate these documents, but before doing so, IE tools require training examples. These examples are normally created manually by human annotators. Currently, there exist very few tools designed to support such people. This paper proposes a methodology aimed at supporting annotators by reducing the number of annotations required by an IE system therefore having effective learning. The whole methodology is implemented in the Melita system which will also be described in this paper. Finally enhancements to the existing methodology are being proposed in order to make IE accessible to a wider range of users, from inexperienced to expert users.

1 Introduction

Many application fields such as the Semantic Web and Knowledge Management require as a precondition the ability to semantically annotate texts and most of the recent methodologies need manual annotation of texts. Manual document annotation is difficult, error prone and sometimes impossible to perform by the common users. For this reason there exists a growing necessity of automated support for document annotation. Information Extraction (IE) can be used as a support for annotation, however most of the current IE technologies require skilled human effort to port the IE to new domains (e.g. Natural Language Processing (NLP) experts). The use of Machine Learning for IE called Adaptive IE (AIE)(Califf et al., 1999)(Basili et al., 2000)(Ciravegna et al., 2001) can be exploited to allow nave users (i.e. users knowledgeable about their domain but having limited knowledge when it comes to computing) to port IE systems so as to avoid referring to NLP experts.

In this paper we propose a framework for semi-automatic semantic annotation of documents together with an overview of future enhancements. In Section 2 we have a brief look at the existing annotation tools available. Section 3 presents our methodology as implemented in the Melita system (Section 3.1). Before we conclude, in Section 4, we will throw light on how we intend to enhance the Melita system in order to give more power to the user without sacrificing its ease of use.

2 Related Work

This section highlights various existing systems that contributed towards adding annotations to web text. Systems such as Annotea (Kahan and Koivunen, 2001), ComMentor (Rscheisen et al., 1994), CoNote (Gay et al., 1999),CritLink[1], iMarkup[2] and Yawas (Denoue and Vignollet, 2000) are similar in spirit to our system but they propose to solve a different problem from the one we are attempting to tackle since they tackle the task of generic annotation by empowering users with tools capable of adding comments to any web site. In our case the use of annotation is mainly restricted towards training an IE

[1] http://crit.org/ ping/ht98.html
[2] http://www.imarkup.com/

system. Annotation is considered as a learning process whereby the annotator is teaching the IE system what concepts are required for annotation.

Our approach is more similar to systems such as GATE annotation tool(Maynard et al.,), MnM (Domingue et al., 2002), the Alembic Workbench(David et al., 1997) and Ontomat(Handschuh et al., 2002). All of these systems support annotation so as to train an IE system. The general approach taken is to help the user in creating a set of test documents using simple techniques like automatically marking co-occurrences of marked concepts or using more sophisticated techniques like pre-annotating the documents using hand crafted rules. A batch of annotated documents is normally given to the IE algorithm for learning and the cycle continues until the learning algorithm has seen enough examples to cover most of the concepts in the domain. Although these systems offer some support to the users annotating, they do so in a very light way. All systems ask the user to annotate most of the documents in the training collection but most of the time, an IE engine is capable of learning with only a fraction of those documents (provided they are a good representation for the domain). In all methodologies, the learning and testing is part of the main annotation cycle. Therefore the users at times may need to stop in order to train the IE system. Finally all systems except for Alembic do not make use of previous annotations to bootstrap further annotations. This technique is referred to as active learning and has been proven to reduce the burden of manual annotation up to 80% in some cases (Thompson et al., 1999).

3 Methodology

The proposed methodology aims to gradually change the traditional role of the user from one of annotator to one of supervisor. It does so while catering for three important factors timeliness, intrusiveness and effectiveness. The first shows the ability to react to user annotation: how timely is the system to learn from user annotations. The second represents the level to which the system bothers the user, because for example it requires CPU (and therefore stops the user). The third one refers to the ability of exploiting effectively all the information available in order to reduce annotation to the maximum.

The method proposed begins in a way similar to traditional annotation tools i.e. the user is asked to annotate a document according to some concepts defined in an ontology. What differs from traditional approaches is that after the first document is tagged the user does not need to train the IE algorithm but is immediately asked to annotate another document. Without the user noticing it, the system sends the annotated document to the learning algorithm for training. Before training, the IE engine keeps note of the user's annotations, removes them from the document and tries to re-annotate the document. The annotations obtained from the IE engine are compared with the original ones from the user and the precision level of the algorithm is calculated based on the number of tags matched. This cycle continues until the algorithm reaches a level of precision above a minimum threshold set by the user. At this stage the system suggests annotations, the user stops annotating and starts supervises the automatic annotations of the system. Eventually, the annotations of the system reach such a high level of precision that the annotation process stops because the IE engine covers most of the domain. The following section will describe how this methodology was implemented.

3.1 The Melita System

Melita is a semi-automatic annotation tool that has AIE integrated in it and supports the users in the process of annotation. It demonstrates how a typical annotation interface could interact with the IES. The novelty of Melita is the possibility of tuning the AIE system so as to provide the desired level of pro-activity and intrusiveness provided by the IE engine. It also allows smart sorting and scheduling of texts that will result in effective learning. At the heart of Melita, the AIE tool Amilcare(Ciravegna, 2001a) is being used. Since Amilcare's approached proved to be one of the best available in different tests(Ciravegna, 2001b); it was chosen as the main AIE algorithm.

Pro-activity and timeliness are catered for in Melita in various ways. To begin with the methodology mentioned above is fully implemented using a client/server approach. As soon

as a user annotates a document, this document is sent to the learning algorithm which lies on a server (either local or remote). The learning algorithm independently is always run as a background process to make sure that no resources are taken from the user. The system is pro-active in the sense that it does not wait for the user to learn and calculate statistics. It takes the initiative to do any pre-processing which will be used in future. This goes hand in hand with timeliness because information is processed immediately even if there's no need for it at present. It exploits every opportunity to make the most of the resources available at that current point in time.

Intrusiveness is handled by Melita in several ways. To begin with, a button on the main interface is used to stop the system from intruding by blocking any suggestions from the learning algorithm. If the suggestion button is on, then the user will receive suggestions from the system, but not all suggestions are displayed to the user since some suggestions (especially at the start of the session) may have very low precision or recall. To allow the user to restrict the suggestions accepted, a component having two movable knobs is displayed for each concept. The knobs can be moved and their position is equivalent to a balance between precision and recall also called the f-measure (where 0% indicates low precision and low recall, while 100% equals high precision and high recall). The lower knob is the suggestion knob while the higher knob is the certainty knob. Rules whose f-measure is below the suggestion knob are not displayed while rules above the suggestion knob but below the certainty knob are displayed as suggestions (in Melita suggestions are shown using a coloured border around the target concept and they must be validated by a user before they are accepted). Rules above the certainty knob are certain to be correct and are displayed using a filled coloured square around the target concept. Using this approach intrusiveness is tuned by the user according to user's requirements without him actually knowing that he is tempering with precision and recall.

Effectiveness is achieved through the document sorting mechanism. This approach dynamically sorts the documents after every annotation in order to find the document that best covers the unexplored areas of the domain[3]. Documents are rated according to the number of tags automatically found by the IE engine. The document with the least number of tags is chosen for annotation because it is the document from which the learning algorithm can learn new rules if it is annotated. This approach has led to a quicker convergence of the learning algorithm whilst overcoming the problem of data sparseness.

In several experiments we conducted, Melita produced quite astonishing results improving the performance of the IE engine. In most of the concepts it achieves an f-measure of 82% after 10 documents. A detailed summary of the experiments can be seen in (Ciravegna et al., 2002).

4 Next Generation Adaptive IE Systems

The methodology presented proved to be quite successful both in the experiments we performed and also when it was used to annotate a number of domains. From these experiences it seems that the way forward is not to consider annotation and IE as two separate and distinct tasks but in order to gain the most benefits they must work hand in hand. The next generation prototypes we are constructing fuse AIE technologies and annotation interfaces and make use of our methodology to gain the maximum benefit.

The methodology we proposed in this document was quite generic and with it, we targeted any kind of user, but the reality is that there are several categories of users all with different backgrounds and needs. Because of this, we identified three distinct category of users and our new prototypes will be designed to cater for their individual needs. We also acknowledge the fact that users may not fall in exactly these categories but somewhere in between. Therefore the system will make sure that a user form one category can use tools designed for other categories. The main categories are:

4.1 Naive Users

These users are knowledgeable about their domain but have limited knowledge when it comes

[3]We assume there are no irrelevant documents in the collection since irrelevant documents can be filtered out beforehand.

to computing. In order to set up the system they only need to specify an ontology and a corpus of documents. Apart from this, all that is required from the user to use the system is the ability to highlight concepts in the document according to concepts in the ontology. The system will also offer advanced features disguised as simple widgets like the component that tunes precision and recall (See Section 3.1). By using this component the user will see tags being updated in real time according to the movement of the knob and the calibration of the component stops when the results are satisfactory.

A potential of this kind of user is domain knowledge. In order to exploit this, the system will highlight words found around the concept that are part of the rules induced by the IE engine. These words will be highlighted using a slightly lighter colour than the highlight of the main concept, to show they are used to help identify the concept. The user will be able to remove or add such highlights. By doing so the user will be unconsciously guiding the algorithm to use certain cue words which are good at identifying the current concept. Therefore the algorithm will induce rules faster because it can use heuristics indirectly provided by the user. It will also converge faster because it is being guided by the domain knowledge of the user.

4.2 Application Expert

This person is at a level between nave and expert user. He is capable of tuning the application but does not have the expertise of an IE expert. Our tool will allow him to have access to advanced features such as adjusting parameters like pattern length used in the learning phase, setting the precision and recall levels explicitly and access to other internal settings.

Due to the fact that this kind of user will have limited knowledge about the nature of information but also considering that his role is to maximise the potential of the IE engine, it is imperative that he is allowed to tweak around with rules in the simplest way possible. The system provides an abstraction for rules in the document being edited. It highlights the words which form part of a rule using a slightly lighter colour than the highlight of the concept they are associated with. When a word is selected by the user a percentage is shown indicating the level of generalisation of that word used by the rule. 0% indicate that a very specific level is used (i.e. the word is explicitly part of the rule) and 100% indicate the most generic level. By sliding the percentage bar the user will be able to see the effect of those changes in real time through components which graphically show precision, recall, f-measure and error levels of the new rule. The highlights are not fixed and the user can add or remove any of them. This tweaking with the rules will allow the user to change rules without the need of understanding linguistic properties.

4.3 IE Expert

Our final user is the IE expert who knows how to control an IE engine and wishes to extract the maximum power from the AIE system. All the benefits offered to the other type of users will be available but this user will require more sophisticated tools. Therefore, the system has a fully fledged rule editing environment.

Using this environment a user can view a list of rules together with statistics about every rule. In this view rules can be compared simultaneously. This is done in order to help the user restructure groups of rules. For example, it may be the case that a group of similar rules can be compressed together in one rule. This view will also enable easy browsing and selection of individual rules. Once a rule is identified as requiring change or even if the user would like to create a new rule, the rule is opened in the rule editor. This program presents to the user several facilities in order to allow him to develop the best rule possible. He has the faculty to change individual properties of a rule and also insert new ones. The changes on the rule can be tested immediately in real time and all the examples in the text where the rule applies are presented to the user. Based on previously annotated documents the system displays positive and negative examples covered by the rules. The system will also display examples of where the rule fires in the test corpus (which is untagged). At this stage the user can separate the positive from the negative results and using this new knowledge the system induces a new rule which is presented to the user for verification. The cycle continues until the user is satisfied with the new rule.

What is actually happening in the rule editing environment is that we are again using our

methodology but this time at a deeper level in order to help the user create handmade rules quickly. The system will be guiding the user towards creating effective rules both by suggesting new rules and by providing important statistical results .

5 Conclusion

The prototypes we presented will slowly lead to a transition from plain annotation to modifying rules. This methodology is innovative because the focus is being transferred from creating powerful tools usable only by experts to developing tools usable by many, yet preserving the power of previous tools. To our knowledge no-one has ever done this before and we believe that it is this usability issue that has been the barrier between modern technology and users. Our research focuses on different kind of tools which are aimed at a wide range of users but mainly to those living in the suburbs of the Information Society.

References

R Basili, F Ciravegna, and R Gaizauskas, editors. 2000. *Workshop on Machine Learning for IE*, Berlin. ECAI2000.

M E Califf, D Freitag, N Kushmerick, and I Muslea, editors. 1999. *Workshop on Machine Learning for Information Extraction*, Orlando, Florida, July. AAAI-99.

F Ciravegna, N Kushmenrick, R Mooney, and I Muslea, editors. 2001. *Workshop on Adaptive Text Extraction and Mining held in conjunction with the 17th International Conference on Artificial Intelligence*, Seattle, August. IJCAI-2001.

F Ciravegna, A Dingli, D Petrelli, and Y Wilks. 2002. User-system cooperation in document annotation based on information extraction. In *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02)*, October.

F Ciravegna. 2001a. Adaptive information extraction from text by rule induction and generalisation. In *17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, August.

F Ciravegna. 2001b. $(lp)^2$, an adaptive algorithm for information extraction from web-related texts. In *IJCAI-2001 Workshop on Adaptive Text Extraction and Mining held in conjunction with the 17th International Conference on Artificial Intelligence*, August.

D David, J Aberdeen, L Hirschman, R Kozierok, P Robinson, and M Vilain. 1997. Mixed-initiative development of language processing systems. In *Fifth Conference on Applied Natural Language Processing*, pages 348–355, April.

L. Denoue and L. Vignollet. 2000. An annotation tool for web browsers and its applications to information retrieval.

J B Domingue, M Lanzoni, E Motta, M Vargas-Vera, and F Ciravegna. 2002. Mnm: Ontology driven semi-automatic or automatic support for semantic markup. In *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02)*, October.

Geri Gay, Amanda Sturgill, and Wendy Martin. 1999. Document-centered peer collaborations: An exploration of educational uses of networked communication technologies. *Computer Mediated Communication*, 4.

S Handschuh, S Staab, and F Ciravegna. 2002. S-cream - semi-automatic creation of metadata. In *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02)*, October.

Jose Kahan and Marja-Ritta Koivunen. 2001. Annotea: an open RDF infrastructure for shared web annotations. In *World Wide Web*, pages 623–632.

D Maynard, V Tablan, H Cunningham, C Ursu, H Saggion, K Bontcheva, and Y Wilks. Architectural elements of language engineering robustness. *Natural Language Engineering – Special Issue on Robust Methods in Analysis of Natural Language Data*, 2002.

Martin Rscheisen, Christian Mogensen, and Terry Winograd. 1994. Shared web annotations as a platform for third-party value-added information providers: Architecture, protocols, and usage examples. *Technical Report, Computer Science Department, Stanford University.*

C. A. Thompson, M. E. Califf, and R. J. Mooney. 1999. Active learning for natural language parsing and information extraction. In *Sixteenth International Machine Learning Conference (ICML-99)*, pages 406–414, June.