Automatic Multi-Layer Corpus Annotation for Evaluating Question Answering Methods: CBC4Kids

Jochen L. Leidner   Tiphaine Dalmas   Bonnie Webber   Johan Bos   Claire Grover
Institute for Communicating and Collaborative Systems (ICCS),
School of Informatics, University of Edinburgh,
2 Buccleuch Place, Edinburgh EH8 9LW, Scotland, UK.
jochen.leidner@ed.ac.uk

Abstract

Reading comprehension tests are receiving increased attention within the NLP community as a controlled test-bed for developing, evaluating and comparing robust question answering (NLQA) methods. To support this, we have enriched the MITRE CBC4Kids corpus with multiple XML annotation layers recording the output of various tokenizers, lemmatizers, a stemmer, a semantic tagger, POS taggers and syntactic parsers. Using this resource, we have built a baseline NLQA system for word-overlap based answer retrieval.

1 Introduction

Linguistic corpora marked up with XML represent the state of the art in language engineering. Recently, reading comprehension tests have received increased attention for testing Question Answering methods. We present our ongoing project to develop a re-usable resource for reading comprehension and Natural Language Question Answering (NLQA) that we hope will be useful as a controlled test-bed for developing and evaluating robust NLQA methods. Starting from MITRE's CBC4Kids corpus collection (Breck et al., 2001), we have created a practical multi-layer annotation scheme and added various strata of linguistic annotation automatically [1] using state-of-the-art NLP tools. This paper presents the architecture, the various tool sets we used and the distributed development scenario we worked in. We also describe how the chosen multi-layer scheme naturally leads to a simple implementation of our baseline question answering system and an evaluation program.

2 Automatic Linguistic Annotation

2.1 Distributed XML development scenario

Our distributed development scenario is shown in Figure 1. A normalization phase of the corpus produces valid XML. After this, development team members each applied the same process to each NLP tool assigned whose output was desired as an annotation layer: a wrapper was created to convert XML into the tool's input format, and another wrapper to convert the tool's output back into a well-formed XML stratum that could be inserted in the XML stream on the fly. This distributed form of collaboration easily scales up to larger development teams, where individual team members are free to choose different implementation languages and glue mechanisms. The final document instance trees were then merged. While a generic XSLT tree union script can be used for this, we instead defined one tree to be the master instance and added all new subtrees present in the second instance. The result was validated against the DTD and transformed further.

2.2 Design principles

While our work is driven by the observation (Cotton and Bird, 2002): "With all the annotations expressed in the same data model, it becomes a

[1] i.e. there is no gold standard for the linguistic annotation, but task-based "gold answers".

straightforward matter to investigate the relationships between the various linguistic levels. Modeling the interaction between linguistic levels is a central concern", we do not aspire to create a new reference annotation model for this type of corpus, but rather to develop a reusable data resource. The original CBC4Kids corpus was developed at MITRE[2], based on a collection of newspaper stories for teenagers written for CBC's Web site[3]. To each article selected for inclusion in the corpus, the MITRE group added a set of 8-10 questions of various degrees of difficulty. The corpus also includes one or more answers for each question, in the form of a disjunction of one or more phrases (the 'answer key'). Due to the wide availability of XML processing tools, we decided to define an XML DTD for the CBC4Kids corpus and to convert various linguistic forms of annotation into XML and integrate them so as to provide a rich knowledge base for our own NLQA experiments and potential re-use by other groups. We selected a set of tools with the guiding principles of 1) public availability, 2) usefulness for the replication of our baseline system, and 3) quality of the automatic annotation. Because most available tools (with the exception of LT TTT, (Grover et al., 2000)) do not output XML, we had to develop a set of converters.

2.3 Linguistic layers

Each sentence has three different representations: 1) the original string, 2) a list of tags labeled TOKEN encoding the results from linguistic tools that output lexical information (POS tags, stems, etc.), 3) a list of trees (PARSEs) corresponding to analyses at a non-terminal level, i.e. syntactic or dependency graphs. This is a compromise between minimizing redundancy and maximizing ease of use. In particular, there is no link between token positions and the corresponding occurrences of words in the parse trees/dependency graphs. Any annotation scheme with a tighter coupling would require an alignment step which, in many cases, would have to remain incomplete due to idiosyncrasies of the tools: for instance, a parser that used its own built-in tokenization might yield a different number of tokens from `tokenizer.sed`.[4]

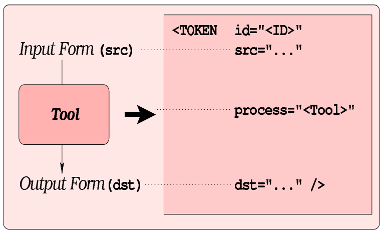
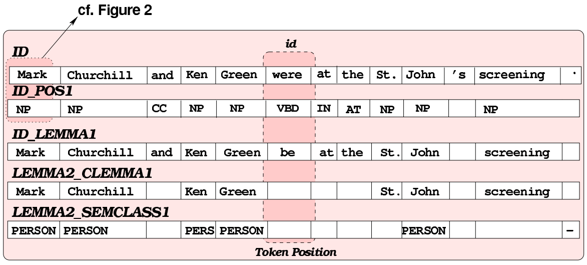


Figure 2: Building a new layer of TOKEN tags.



ID												
Mark	Churchill	and	Ken	Green	were	at	the	St.	John	's	screening	.
ID_POS1												
NP	NP	CC	NP	NP	VBD	IN	AT	NP	NP		NP	
ID_LEMMA1												
Mark	Churchill	and	Ken	Green	be	at	the	St.	John		screening	
LEMMA2_CLEMMA1												
Mark	Churchill		Ken	Green				St.	John		screening	
LEMMA2_SEMCLASS1												
PERSON	PERSON		PERS	PERSON					PERSON			–

Figure 4: Multiple annotation layers.

Because various forms of linguistic processing depend on the output of other tools, we wanted to make the processing history explicit. We devised a multi-layer annotation scheme in which an XML `process` attribute refers to a description of the input (token or tree), the output, and the tool used. Figure 2 shows how a TOKEN layer is built. This annotation allows for easy stacking of mark-up for tokenization, part-of-speech (POS) tags, base forms, named entities, syntactic trees etc. (Figure 4). The word-form token from Figure 2 are then repeated in the PARSE trees (`<LEAF type="Scotia"/>`).

Figure 5 and Figure 6 show the current status of our annotation "pipe tree" for tokens and trees/graphs, respectively, as described below.[5] Figure 3 gives an overview of our current annotation layers. A comprehensive description of the tools and structure can be found in the manual (Dalmas et al., 2003a) distributed with the corpus. This procedure is carried out for the stories, questions, and answer keys (Appendix B).

[2] The contact person for the corpus is Lisa Ferro (lferro@mitre.org).

[3] http://www.cbc4kids.ca/

[4] Treatment of *doesn't* as *does n't* is but one example.

[5] We call it a "pipe tree" because it represents a set of "pipe lines" (like UNIX pipes) with common initial sub-steps.

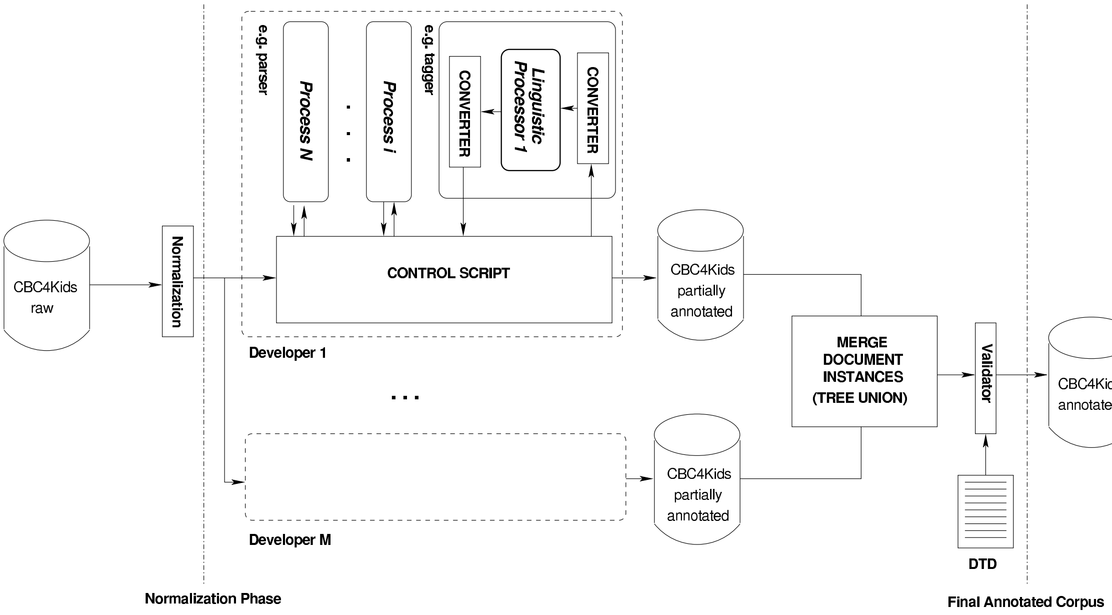


Figure 1: Building a richly annotated corpus in a distributed XML scenario.

Type	Tool	Process ID	Reference
Sentence Boundaries	MXTERMINATOR	`ID`	(Ratnaparkhi, 1996)
Tokenization	Penn tokenizer.sed	`ID_TOK1`	
	Tree-Tagger (internal)	`ID_TOK2`	(Schmid, 1994)
	LT TTT	`ID_TOK3`	(Grover et al., 2000)
Part-of Speech	MXPOST	`TOK1_POS2`	(Ratnaparkhi, 1996)
	Tree-Tagger	`TOK2_POS1`	(Schmid, 1994)
	LT POS	`TOK3_POS3`	(Mikheev et al., 1999)
Lemmatization	CASS ‘stemmer’	`TOK1_LEMMA2`	(Abney, 1996)
	Tree-Tagger	`TOK2_LEMMA1`	(Schmid, 1994)
	morpha	`POS1_LEMMA3`	(Minnen et al., 2001)
Stemming	Porter stemmer	`LEMMA2_STEM1`	(Porter, 1980)
Stop-Word Filtering	Deep Read	`LEMMA2_CLEMMA2`	(Hirschman et al., 1999)
	Deep Read	`STEM1_CSTEM1`	(Hirschman et al., 1999)
Syntactic Analysis	Apple Pie Parser	`POS2_SYN1`	(Sekine and Grishman, 1995)
	Minipar relations	`TOK1_SYN2`	(Lin, 1998)
	CASS chunk trees	`POS1_SYN3`	(Abney, 1996)
	CASS dependency tuples	`POS1_SYN4`	(Abney, 1996)
	Collins parse trees	`POS2_SYN5`	(Collins, 1997)

Figure 3: Annotation tools: current layers.

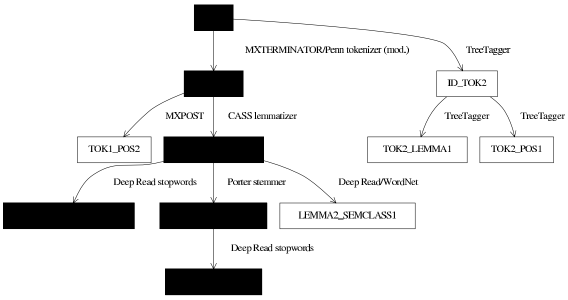


Figure 5: Annotation layers per token. The replicated Deep Read baseline system pipeline is highlighted.

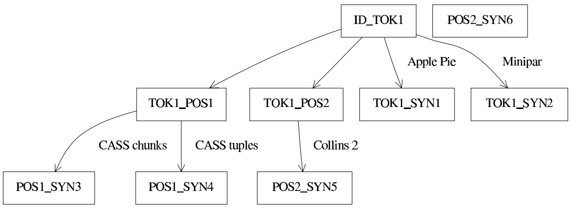


Figure 6: Annotation layers per sentence.

2.4 Normalization and annotation of the corpus

The original CBC4Kids corpus from MITRE comes marked up with XML-like tags. After authoring the Document Type Definition (DTD), processing comprised two steps (Figure 1): first, we normalized the corpus so as to make sure the data consistently fits the form described by our DTD. Because the minimal scheme requires sentence boundary detection (cf. Section 2.5) and the original CBC4Kids corpus only contained markup for paragraphs, normalization also involved splitting each paragraph into a list of sentences. Secondly, we enriched the corpus with linguistic annotation layers. Annotation layers are optional in our DTD. Each annotation layer is added by a program taking an XML file as input and ouputing another XML file containing the additional layer; Figure 2 shows the internal process.

2.5 Description of the layers

Sentence boundary detection. We used MXTERMINATOR (Ratnaparkhi, 1996) to split each paragraph into sentences. Questions and human answers are already demarcated in the source CBC corpus released by MITRE .

Tokenization. Each sentence was tokenized using the Penn Treebank tokenizer, a **sed(1)** script by Robert MacIntyre (University of Pennsylvania)[6]. We modified it slightly before running it on the corpus so as to recognize number separators, URLs, and intra-sentential quotations which were characteristic of the CBC corpus. The resulting token sequence was defined as process `ID_TOK1`.

POS Tagging. We recorded the results of two POS taggers for comparison purpose: TreeTagger and MXPOST. TreeTagger (Schmid, 1994) is a POS tagger based on decision tree induction. Trained and tested on a Penn Treebank sample, it has a reported accuracy of 96.34% (trigram maximal window size). The POS tags of TreeTagger define our layer `TOK2_POS1` (TreeTagger comes with a built-in tokenizer). MXPOST is a POS tagger based on the Maximum Entropy framework that has a reported accuracy of 96.6% on Wall Street Journal text (Ratnaparkhi, 1996). We have added a token layer `TOK1_POS2` based on MXPOST.

Lemmatization. `TOK1_LEMMA1`, our first token layers of lemmata, is provided by TreeTagger. Additionally, we obtained a second baseform layer `TOK1_LEMMA2` using the rule-based program `stemmer` from the CASS software distribution (Abney, 1996).[7]

Stop-Word Filtering. We used the same stopwords set as described in the Deep Read baseline (Hirschman et al., 1999).

Stemming. (Porter, 1980) describes a simple stemmer for English (`LEMMA2_STEM1`).

Semantic Classes. Deep Read (Hirschman et al., 1999) uses WordNet to check words for subsumption of the synsets PERSON and/or LOCATION. We have integrated the result of this lookup as layer `LEMMA2_SEMCLASS1`.

Parsing. `PARSE` tags record the output of several different parsers that we have included in our pipe trees.

The Apple Pie Parser (APP) is a statistical parser trained on the Wall Street Journal subset of the Penn Treebank (Sekine and Grishman, 1995).

[6] `http://www.cis.upenn.edu/treebank/tokenization.html`

[7] Despite the name, `stemmer` is a lemmatizer rather than a stemmer.

It comprises only non-terminals for NP and S and parses by re-combination of NP/S-fragments, memo-izing the complete training set. On unseen WSJ material, it has been reported to achieve a labeled precision[8] of 72.61% and 83% for sentences up to 15 words. APP returnes a single best tree, which we have incorporated as process POS2_SYN1.

Minipar (Lin, 1998) is a rule-based parser that implements a procedural model of Government & Binding (GB) realized as message passing; it is derived from Principar (Lin, 1995), incorporates knowledge about named entities and comprises a lexicon ≈90k lemmata. Its output consists of dependency relations over word token positions. It has a reported labeled dependency precision of 88.54%. This is the process TOK1_SYN2.

Shallow processing techniques have emerged as an efficient way to deal with large quantities of text. 'Chunking' – partial parsing by iterative bottom-up bracketing using multi-layer deterministic finite-state transducers for non-recursive noun/verb groups ('chunks') – has been described by Abney (Abney, 1996) and implemented in his CASS. The chunker outputs either trees or dependency relation tuples.[9] We define a layer POS1_SYN3 with trees of CASS chunks and POS1_SYN4 with the dependency tuples.

(Collins, 1997) presents three statistical, lexicalized parsing models. We chose his model 2 (which models left and right dependents), for integration as layer POS2_SYN5. The POS2 layer is used as input because Collins' parser uses MXPOST POS tags for handling unknown words. Collins reports 88.35% labelled precision for this model on sentences with less than 40 words. The average sentence length in CBC4Kids is 18 words (maximum 57).

The layers described here allow detailed comparisons of components' contribution for any NLQA method by exploring different paths in the annotation "pipe tree".

This annotation is work in progress and we are planning to include further layers featuring analyses of LT TTT, LT POS, LT CHUNK, named entity annotation using MITRE's Alembic (Aberdeen et al., 1995), LTG's MUC-7 system (Grover et al., 2000), as well as anaphora reference information.

3 Building NLQA Systems as Set of XML Filters

This section describes the architecture of our question answering system. We built it to transfer the baseline results from the word overlap method used by the Deep Read system in connection with the REMEDIA corpus (Hirschman et al., 1999) to the annotated CBC4Kids data and to support our investigation of more sophisticated methods.

We exploited our XML annotation scheme using the STEM1_CSTEM1 and LEMMA2_CLEMMA2 layers for a baseline based on content stems and content lemmata, respectively. For the results of our baseline system see (Dalmas et al., 2003b). The layer is a parameter, so any–even a user-defined–layer may be used with our existing implementation. Figure 5 shows these final layers we used and their ancestors in the linguistic pipeline.

We have implemented a batch NLQA system as a set of filters in the functional programming language Haskell.[10]

The XML markup of linguistic information greatly simplified the implementation part: the NLQA system was reduced to a function filtering a tree (the XML document containing story and questions) and computing intersection (overlap) on lists of tokens. Figure 7 shows the root of the XML tree structure of a CBC4Kids document. A document (DOC) instance comprises the story and the associated set of questions and answers. Question Answering is reduced to selecting a desired layer and intersecting the bags of tokens associated with questions and answers, respectively.

4 Related Work

Corpus annotation. Some other corpora have multiple annotation layers. For example, (Grover et al., submitted) use five linguistic layers for their DISP corpus of biomedical abstracts in order to

[8] The reported numbers for parse tree evaluation refer to the PARSEVAL.

[9] Since the latter output format is based not on token-position but on the surface string of the region, there is a potential for ambiguity if a surface string occurs multiple times in the same sentence.

[10] For an introduction, see (Thompson, 1999) or http://www.haskell.org/.

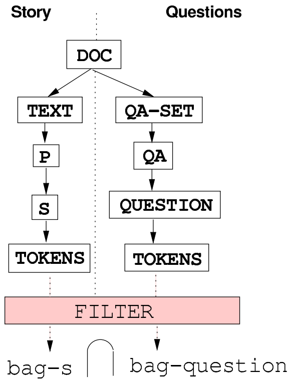


Figure 7: Interplay between annotation layers and our NLQA system. White boxes represent XML tags.

compare the relative utility of shallow versus deep parsing in analyzing nominal compounds.

XML alternatives. In the TIPSTER project (NIST, online), a different architecture for text processing was proposed that does not make use of XML: a TIPSTER-compliant application annotates text by maintaining character offset pairs indicating the beginning and end of the zone together with the type of token, phrase etc.

Work on pipelines. An XML pipeline can also be described in a special glue language for streams such as the XML stream processor STnG (Krupnikov, submitted). The main advantage would be to formulate the processing pipeline in a language that allows any kind of executable to be called without using a combination of XML parser and a programming language (such as LTG `xmlperl` plus Perl).

XML-aware languages. Finally, new special-purpose programming languages are already being designed, which—like CDuce (Benzaken et al., 2002)—treat DTDs and their elements as first-order objects and allow direct manipulation of DTDs and XML document instances within the functional paradigm; these are expected to simplify XML processing further.

5 Lessons Learnt and Future Work

XML Pervasiveness. The NLP community has now widely adopted the use of SGML or XML for computer corpus annotation. XML-aware software such as input/output application programming interfaces (APIs), search and transformation tools are now also available. However, the linguistic community has not generally adopted XML as the standard output format for parsers, taggers etc., so that it is still necessary to invest significant time to develop converters. The collaborative development scenario we have used here has proven effective in supporting this in a distributed fashion. Because there are is a large number of tools available for XML processing and it is programming language independent, XML is the ideal corpus exchange meta-format within and between groups.

DTD. Tokenization is currently considered a standard word-based process. In fact, it should also be encoded as a non-terminal layer because its original input is a string, from which different token sequences (with varying lengths) can be extracted, depending on the individual tool (see footnote 3).[11]

Applications. We are planning to use the CBC4Kids for future NLQA experiments as a testbed for evaluation. The present parse trees and dependency relations will allow us to develop semantically oriented answering strategies, including shallow inference (Webber et al., 2002), and performance measurements based on output from different sets of tools can be compared in a task-based evaluation.

Re-use. We do not know of any other corpus that has been automatically annotated with comparably rich strata of linguistic knowledge and believe that the corpus can be a valuable resource for other NLQA research groups as well. The annotated corpus will be distributed by MITRE with layers as given above, including answers given by our system for the Deep Read baseline. Please contact Lisa Ferro directly for a copy.[12]

Acknowledgments. Lynette Hirschman and Lisa Ferro at MITRE provided us with the initial CBC4Kids corpus. We would also like to thank the authors of all the tools we have mentioned this paper for making them available to the academic community. Thanks to Maria Lapata, Dekang

[11] Parallel readings on the tokenization level are generally not followed up for efficiency reasons.

[12] `lferro@mitre.org`

Lin, Katja Markert, Satoshi Sekine, Bill Wellner and two anonymous reviewers for advice and feedback. We would like to acknowledge the financial support of the German Academic Exchange Service (DAAD) under grant D/02/01831, of Linguit GmbH (research contract UK-2002/2), and of the School of Informatics, University of Edinburgh.

References

J. Aberdeen, D. Day, L. Hirschman, P. Robinson, and M. Vilain. 1995. MITRE: Description of the Alembic system used for MUC-6. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pages 141–155.

S. Abney. 1996. Partial parsing via finite-state cascades. *Journal of Natural Language Engineering*, 2(4):337–344.

V. Benzaken, G. Castagna, and A. Frisch. 2002. CDuce: A white paper. In *PLAN-X: Workshop on Programming Language Technologies for XML*.

E. Breck, M. Light, G. Mann, E. Riloff, B. Brown, P. Anand, M. Rooth, and M. Thelen. 2001. Looking under the hood: tools for diagnosing your question answering engine. In *ACL-2001 Workshop on Open-Domain Question Answering*.

M. J. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the ACL*, Madrid. Association for Computational Linguistics.

S. Cotton and S. Bird. 2002. An integrated framework for treebanks and multilayer annotations. In *Proceedings of the Third International Conference on Language Resources and Evaluation*.

T. Dalmas, J. L. Leidner, B. Webber, C. Grover, and J. Bos. 2003a. *Annotating CBC4Kids: A Corpus for Reading Comprehension and Question Answering Evaluation. (Technical Report)*. School of Informatics, University of Edinburgh.

T. Dalmas, J. L. Leidner, B. Webber, C. Grover, and J. Bos. 2003b. Generating annotated corpora for reading comprehension and question answering evaluation. In *EACL-2003 Workshop on Natural Language Processing (NLP) for Question-Answering*.

C. Grover, C. Matheson, A. Mikheev, and M. Moens. 2000. LT TTT–a flexible tokenisation tool. In *Proceedings of Second International Conference on Language Resources and Evaluation (LREC 2000)*.

C. Grover, M. Lapata, and A. Lascarides. submitted. A comparison of parsing technology for the biomedical domain. In *Journal of Natural Language Engineering*.

L. Hirschman, M. Light, E. Breck, and J. D. Burger. 1999. Deep Read: A reading comprehension system. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*.

K. A. Krupnikov. submitted. STnG: a streaming transformation and glue engine for XML. In *Proceedings of XML Europe 2003*.

D. Lin. 1995. A dependency-based method for evaluating broad-coverage parsers. In *Proceedings of IJCAI-95*.

D. Lin. 1998. Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*.

A. Mikheev, C. Grover, and M. Moens. 1999. XML tools and architecture for named entity recognition. *Journal of Markup Languages: Theory and Practice*, 3:89–113.

G. Minnen, J. Carroll, and D. Pearce. 2001. Applied morphological processing of English. *Journal of Natural Language Engineering*, 7(3):207–223.

NIST. online. *TIPSTER Text Architecture Concept*. http://www-nlpir.nist.gov/related_projects/tipster/.

M. F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

A. Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference*.

H. Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. *International Conference on New Methods in Language Processing*.

S. Sekine and R. Grishman. 1995. A corpus-based probabilistic grammar with only two non-terminals. In *Proceedings Fourth International Workshop on Parsing Technologies*.

S. Thompson. 1999. *Haskell: The Craft of Functional Programming*. Addison-Wesley, Reading, MA, 2nd edition.

B. L. Webber, C. Gardent, and J. Bos. 2002. Position statement: Inference in question answering. In *Proceedings of the LREC Workshop on Question Answering: Strategy and Resources*, Las Palmas, Gran Canaria, Spain.

A Sample Story from CBC4Kids

Mourning in an Alberta Town
April 29, 1999

Pastor Ken Gartly provided comfort and prayer to people in Taber, Alberta yesterday after two students were shot at the town's high school.

Students at W. R. Myers high school had just settled down after lunch when a 14-year-old boy walked in and shot two students, killing one. The shooting comes a week after the shooting tragedy at Columbine high school in Littleton, Colorado. "We have a son and daughter-in-law in Denver," says Pastor Gartly after an evening service at Taber Evangelical Free Church where worshipers discussed the day's tragic events. The dead teenager has been identified as Jason Lang, 17. The other victim, Shane Christmas, also 17, had emergency surgery yesterday at Lethbridge Regional Hospital. This morning his condition was reported as fair to serious. The two grade 11 students were said to be best friends.

Eight thousand people live in Taber, which is 300 kilometres southeast of Calgary. Many members of the community are members of the Mormon Church or are evangelical Christians.

Taber was founded at the beginning of the century. It is mainly made up of decendents of the area's early homestead pioneers, of Central European, Polish, Japanese, Dutch and various other racial backgrounds.
[...]

Police confirmed the gunman was taken into custody by the school resource officer, who is also a member of the Taber Police Service. The six hundred mostly Inuit residents of the northern Quebec village of Kangiqsualujjuaq had planned to bury the bodies of nine of their friends and children in a funeral this afternoon. But the bad weather that resulted in their deaths has also delayed the funeral until Tuesday.

Questions

Who shot two students at a high school in Taber?

What time of day did the Taber shooting take place?

Where is the Taber gunman now?

Who died in the Taber shooting? (see Figure 8)

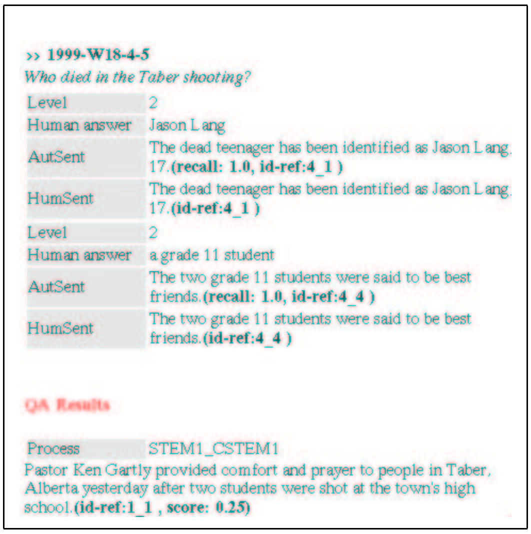
Why is Shane Christmas in the hospital?

Why were yesterday evening's activities at Taber Evangelical Free Church modified?

When was there a school shooting in Colorado?

B The Encoding of Questions and Answers

Figure 10 shows the hierarchical structure of our XML encoding for questions, answers and system results.



>> 1999-W18-4-5
Who died in the Taber shooting?

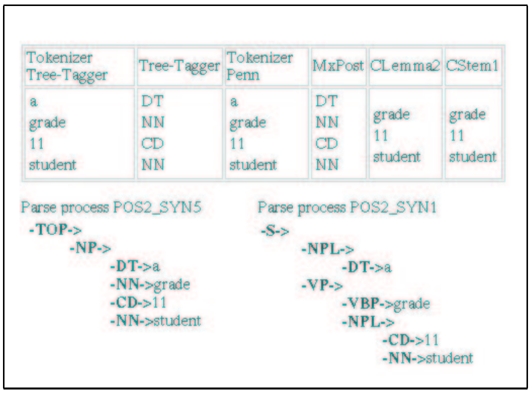
Level	2
Human answer	Jason Lang
AutSent	The dead teenager has been identified as Jason Lang, 17. **(recall: 1.0, id-ref:4_1)**
HumSent	The dead teenager has been identified as Jason Lang, 17. **(id-ref:4_1)**
Level	2
Human answer	a grade 11 student
AutSent	The two grade 11 students were said to be best friends. **(recall: 1.0, id-ref:4_4)**
HumSent	The two grade 11 students were said to be best friends. **(id-ref:4_4)**

QA Results

Process	STEM1_CSTEM1

Pastor Ken Gartly provided comfort and prayer to people in Taber, Alberta yesterday after two students were shot at the town's high school. **(id-ref:1_1 , score: 0.25)**

Figure 8: HTML view of a question. *score* is WdAnsRecall from (Hirschman et al., 1999).



Tokenizer Tree-Tagger	Tree-Tagger	Tokenizer Penn	MxPost	CLemma2	CStem1
a grade 11 student	DT NN CD NN	a grade 11 student	DT NN CD NN	grade 11 student	grade 11 student

```
Parse process POS2_SYN5          Parse process POS2_SYN1
 -TOP->                           -S->
       -NP->                            -NPL->
            -DT->a                            -DT->a
            -NN->grade                  -VP->
            -CD->11                           -VBP->grade
            -NN->student                      -NPL->
                                                    -CD->11
                                                    -NN->student
```

Figure 9: HTML view of some linguistic layers of the corresponding human answer.

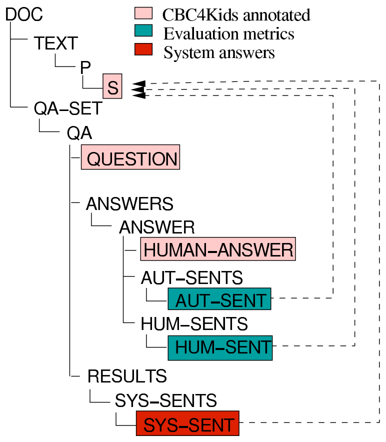


Figure 10: XML tag hierarchy for questions, answers and system results.