A New Approach for Scientific Citation Classification Using Cue Phrases

Son Bao Pham and Achim Hoffmann

School of Computer Science and Engineering
University of New South Wales, Australia
{sonp,achim}@cse.unsw.edu.au

Abstract. This paper introduces a new method for the rapid development of complex rule bases involving cue phrases for the purpose of classifying text segments. The method is based on Ripple-Down Rules, a knowledge acquisition method that proved very successful in practice for building medical expert systems and does not require a knowledge engineer. We implemented our system KAFTAN and demonstrate the applicability of our method to the task of classifying scientific citations. Building cue phrase rules in KAFTAN is easy and efficient. We demonstrate the effectiveness of our approach by presenting experimental results where our resulting classifier clearly outperforms previously built classifiers in the recent literature.

1 Introduction

With an ever increasing number of scientific papers and other documents available, it is impossible for researchers to read everything that might be relevant. It is useful to have a system that assists researchers in dealing with this problem possibly by suggesting only a small number of specific papers that they should read. Building a citation map is a good way of summarizing comments on a paper by other authors. A citation map is a directed graph with papers as nodes. A directed edge between two nodes indicates that one paper cites the other. The type of edges depend on their corresponding citation types. A citation map is useful for experienced researchers as well as for those that are new to the field. The newcomers could use the citation map, for instance, to find the fundamental work in the field. These usually include papers on which others *base* their work. Papers that are *supported* by a large number of other papers are also a good starting point.

Building robust natural language processing systems remains a challenging engineering task. A reason for the persistent difficulty with building more sophisticated NLP systems is the extraordinary variety in which certain content may be presented in natural language. Capturing the vast variety of possible natural language patterns which would allow the same system response, such as classifying entire texts or text segments, appears to require substantial task-specific knowledge.

In this paper, we introduce a new approach for capturing such task-specific linguistic patterns from human experts (most humans who are fluent in a language would qualify as experts).

Our approach builds on the basic idea of Ripple-Down Rules [3], which strongly support the knowledge acquisition in the context of its actual use. For the tasks in natural language processing that means that an expert monitors the system's performance on individual pieces of text. As soon as the system makes a mistake due to some rule *R*, the expert tells the system how the current context, i.e. the given text or text segment, differs from previous contexts in which rule *R* performed satisfactorily. This kind of knowledge acquisition approach proved to be substantially more effective than machine learning techniques applied to the same data [5]. Reason being that the human is in the loop who provides important guidance for building a knowledge base which learning techniques can only achieve by the use of much more data. Since the data needs to be manually classified, the overall involvement of the human can be significantly reduced using a knowledge acquisition approach instead of a learning approach.

We developed KAFTAN, our Knowledge Acquisition Framework for TAsks in Natural language, which is capable of rapidly acquiring cue phrases for classifying individual citations in a text. Recently CiteSeer, a citation visualization tool which performs Autonomous Citation Indexing, has been built [6]. References together with surrounding text are automatically extracted from all papers available to CiteSeer. However, CiteSeer does not provide users with the types of citations between papers. The user has to read the citation context in order to work out the type of citation. KAFTAN would therefore benefit CiteSeer substantially.

2 Related Work

In this section we present the basic idea of Ripple-Down Rules in 2.1 upon which our approach was based. In 2.2 we discuss related work on our application domain of citation classification.

2.1 Knowledge Acquisition with Ripple Down Rules

Ripple Down Rules (RDR) is an unorthodox approach to knowledge acquisition. RDR does not follow the traditional approach to knowledge based systems (KBS) where a knowledge engineer together with a domain expert perform a thorough domain analysis in order to come up with a knowledge base. Instead a KBS is built with RDR incrementally, while the system is already in use. No knowledge engineer is required as it is the domain expert who repairs the KBS as soon as an unsatisfactory system response is encountered. The expert is merely required to provide an explanation for why in the given case, the classification should be different from the system's classification.

This approach resulted in the expert system PEIRS used for interpreting chemical pathology results [3]. PEIRS appears to have been the most comprehensive medical expert system yet in routine use, but all the rules were added by a pathology expert without programming or knowledge engineering support or skill whilst the system was in routine use. The basic idea has been extended into a number of directions - none of it addressing the specific problems present in NLP applications.

Ripple-Down Rules and some further developments are now successfully exploited commercially by a number of companies.

Single Classification Ripple Down Rules A single classification ripple down rule (SCRDR) tree is a finite binary tree with two distinct types of edges. These edges are typically called *except* and *if not* edges. See Figure 1. They are used for evaluating cases and will be discussed later. Associated with each node in a tree is a *rule*. A rule has the form: *if* α *then* β where α is called the *condition* and β the *conclusion*.

Cases in SCRDR are evaluated by passing a case to the root of the tree. At any node in the tree, if the example entails the condition of the node, the node is said to *fire*. If a node fires, the example is passed to the next node following the *except* branch. Otherwise, the case is passed down the *if not* branch. This determines a path through a SCRDR tree for an example. The conclusion given by this process is the conclusion from the last node which fired on this path. To ensure that a conclusion is always given, the root node typically contains a trivial condition which is always satisfied. This node is called the *default* node.

A new node is added to an SCRDR tree when the evaluation process returns the wrong conclusion. The new node is attached to the last node evaluated in the tree. If the node has no exception link, the new node is attached using an exception link, otherwise an *if not* link is used. The case causing the new node being added (call this example e) is associated with the new node and is called the *cornerstone case* for that node. To determine the rule for the new node, the expert formulates a rule which is satisfied by e but not by the cornerstone case for the last node which fired in the evaluation path.

While the process of incrementally developing knowledge bases will eventually lead to a reasonably accurate knowledge base, provided the domain does not drift and the experts are making the correct judgements, the time it takes to develop a good knowledge base depends heavily on the appropriateness of the used language in which conditions can be expressed by the expert. For example, if the target function to be represented is a linear threshold function in the numerical input space and the conditions an expert can use allow only axis-parallel cuts, it will take very long until most of the relevant objects will be classified correctly.

Despite a number of different application areas in which RDR systems have been employed, serious applications in natural language processing have not yet been tackled with Ripple-Down Rules. For natural language tasks it is not obvious what a suitable language for conditions will be. In this paper, we tried rather simple conditions as discussed in 3.2 and they seem to be sufficient for our task.

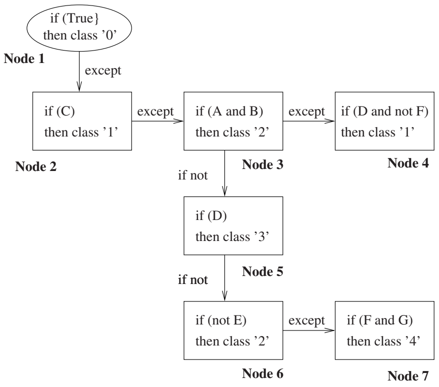


Fig. 1. An example SCRDR tree. Node 1 is the default node. A case for which only 'A' is true is classified as '0' by node 1. If only A, B, and C are true, it is classified as '2' by node 3, while it is classified as '1' by node 4 if on top of A,B, and C also D was true. If only A, C, and D are true it is classified as '3' by node 5. If only A and C are true, the case is classified as '2' by node 6, etc.

2.2 Work on Citation Classification and Linguistic Patterns

Linguistic patterns or cue phrases have a long history as features for determining global importance of the sentences containing them. In Edmundson's approach [2], sentences containing positive cue phrases that express confidence or importance (e.g. *absolutely*, *important*) were candidates for inclusion in the summary. On the other hand, sentences containing stigma words (e.g. *unclear*, *hardly*) were unlikely to be included in the summary. More recent works include Burstein et al. [1] where cue phrases were used to derive rhetorical relations. Teufel [8] also used long cue phrases (e.g. *similar to those of*) for her Argumentative Zoning work.

Nanba and Okumura [7] introduce a supporting tool for writing survey papers. They use cue phrases to classify citations in text into one of three categories i.e type B (Basis), type C (Comparison or Contrast) and type O (other). Inspecting Nanba & Okumura's cue phrases after they were selected manually by using n-gram models suggests that they could easily be expanded to generalize better if our notion of word group were used in those phrases. For example, a sentence having the cue phrase

however, ... they

would be classified into type C. But Nanba & Okumura's list of cue phrases does not contain the phrase

however, ... he/she

which is a similar pattern. This is probably because the latter kind of phrase did not appear often enough in the used corpus (before formulating cue phrases Nanba & Okumura analyzed their corpus by searching for frequently occurring patterns). However, including that phrase would clearly classify more sentences outside the corpus correctly. This problem is addressed in our approach by using word groups which will be described in section 3.4.

Teufel [8] tries to classify sentences into Argumentative Zones which she argues could be used for building citation maps as well as for text summarization.

The seven zones consist of *Background*, *Other*, *Own*, *Aim*, *Textual*, *Contrast*, and *Basis*. Among those, only *Contrast* and *Basis* are relevant to our task.

Different to Nanba & Okumura, Teufel [8] proposes to use meta-discourse features which include formulaic expressions, agent patterns and semantic verb classes. Formulaic expressions are bundled into 20 major semantic groups (e.g. *CONTRAST*, *GAP*, *PROBLEM*, etc.). To determine which group a sentence belongs to, 396 formulaic patterns were manually constructed. An example of a *GAP* formulaic pattern is

as far as @SELF_NOM know

where @SELF_NOM is an entry in a manually constructed concept lexicon consisting of lists of words. Similarly, there are 168 agent patterns and semantic verb classes composed of 365 verbs constructed manually. The patterns manually constructed by Teufel are similar to our patterns used in the condition part of our rules, though Teufel's rules are less expressive. The lexicon is also very much like our word groups.

Both Nanba & Okumura's as well as Teufel's work show that the skillful use of cue phrases can lead to useful tools for purposes such as building citation maps. However, the manual development of suitable cue phrases is obviously a rather time-consuming task. Furthermore, certain cue phrases are bound to contradict each other, i.e. a single sentence will be matched by multiple cue phrases which may well belong to different classes. Resolving such conflicts manually is a time-consuming activity.

Our approach provides automatic support for eliminating such conflicts from the beginning by structuring the knowledge base in a suitable way and by presenting potential conflict-causing sentences to the expert as soon as they are encountered. This makes it very easy for the expert to provide a rule that resolves such conflicts. The integration of the rule into the knowledge base without causing any inconsistencies is also done automatically.

3 KAFTAN Design

Our Knowledge Acquisition Framework for TAsks on Natural language (KAFTAN) allows users to easily and quickly develop a knowledge base for natural language sentence classification. It is in general hard for the expert to formulate a good rule from one example. With KAFTAN, the expert has merely to justify why the current example is of a particular category. The expert will then be guided through a simple process to expand the current rule so that it could cover a larger number of cases. Our KAFTAN system offers the user the following features.

– KAFTAN supports an effective user machine interaction by first highlighting those parts of a sentence that led to a misclassification. Based on that it is

usually easy for an expert to come up with a suitable choice of one or more words in the current sentence that justifies its classification into a different class.
- Following that initial choice of words, KAFTAN offers generalizations of each selected word to a word group composed of synonyms that are appropriate in the context.
- To support the quick selection of synonyms, KAFTAN colour-codes its lists of synonyms according to the various meanings of the original word, which KAFTAN takes from WordNet [4].

3.1 Knowledge Bases Built with KAFTAN

KAFTAN allows users to define a number of classes into which a sentence can be classified. For each class C an SCRDR tree will be acquired which determines whether a sentence is classified as class C or not. Every SCRDR tree is composed of a number of nodes each containing a KAFTAN rule. Rules in KAFTAN are composed of a condition part and a conclusion part. The condition is to be evaluated on an individual sentence. The conclusion part is either a class label or a text segment selector. KAFTAN allows users to build two types of rules. One is for plain sentence classification, i.e. it uses a class label as conclusion. The other type is an information extraction rule, which uses a selector conclusion to select parts of a sentence to be filled into a corresponding field of a template. Both types of rules use the same kind of conditions.

3.2 Conditions in KAFTAN Rules

Regular expression was considered to be used as the condition for KAFTAN rules initially because of its expressiveness. In order to take this advantage however, the user has to predict sentences similar to the one at hand to formulate the desired conditions that cover more cases. This puts a burden on users which is against the motivation of our system. KAFTAN is built to help users minimize their effort in formulating new rules. The user is only required to differentiate the case at hand from the cornerstone cases without worrying about unseen cases. It is therefore desirable to have a condition syntax that is easy and intuitive to use while expressive enough. A simplified version of regular expression turns out to suffice for KAFTAN's objectives.

A condition in KAFTAN is a simple pattern consisting of a sequence of an arbitrary number of word groups (see section 3.4 for details). Between two word groups there may be a gap whose maximum length can be specified (or left unlimited) within the condition. A gap represents effectively a wild-card which is matched by any sequence of words of up to the maximum number of words. Inter-punctuation within a sentence is ignored at this stage. Each word group represents a set of alternative words that may occur in a matching sentence at the respective position. An example condition may be the following:

$(approach|method)\ (gap\ 5)\ (is|was)\ (good)$

This condition would match sentences containing "approach" or "method" followed by "is" or "was" with at most 5 words apart (gap 5), followed immediately by "good". For example, it would match the sentence

Our approach of using knowledge acquisition is good because it makes the development of intelligent systems much easier.

To increase the expressiveness of the condition language, we also allow negation to be used. That is one can require that a particular word, word group or multiple such words or word groups do *not* match the sentence. (This is an important feature when formulating exception rules to a rule.)

3.3 Automatic Check for Consistency

As discussed in section 2.1 our approach has the objective of ensuring the incremental improvement of a system's capabilities. This implies that all sentences which KAFTAN had already classified to the satisfaction of the user need to be classified in the same way after any modification to KAFTAN's knowledge base. The only type of modification of KAFTAN's knowledge base that is allowed is the addition of further (exception) rules to its rule base.

To ensure this and to support the user to add suitable rules, KAFTAN checks automatically for any potential inconsistencies with previous classifications of sentences for each candidate rule a user enters. I.e. KAFTAN stores all sentences it sees along with their classification which was endorsed by the user. For any new rule R a user wants to enter, KAFTAN checks whether R would misclassify any previously classified sentences. If a rule would misclassify such a sentence, this is indicated to the user and KAFTAN asks to modify that rule or to start from scratch to form a new rule. I.e. after a rule has been accepted by KAFTAN, it does not need to be revisited again and will remain in the knowledge base.

3.4 Word Groups

A specific sequence of words used in a condition in a KAFTAN rule would apply to the sentence that triggered the user to enter the rule (the cornerstone case) and may also apply to a number of other sentences. However, in many cases there will be a number of different ways of expressing essentially the same content. Hence, to avoid entering essentially the same rules over and over again with some slight variation in the particular words being used, KAFTAN allows users to use a set of words from which only one has to match a sentence. The set of words can be chosen by the user as needed in order to accommodate synonyms or other related words which are commonly used in place of the particular word in the cornerstone case.

Furthermore, KAFTAN supports the user rather strongly in finding a suitable set of words in the following way:

– WordNet [4] is used to provide an initial set of possible synonyms and hypernyms.

– The synonyms and hypernyms in WordNet are ordered according to different meanings of the query word. These different groups of synonyms and hypernyms are colour-coded when presented to the user.
– The user can form their own sets of related words which can be tailored to suit specific purposes not catered for in WordNet.

For each of the groups *Noun*, *Adjective*, *Verb*, and *Adverb* synonyms in different ”contexts” are highlighted in different colours. If there is no synonym in a particular group, there is no display of that group at all.

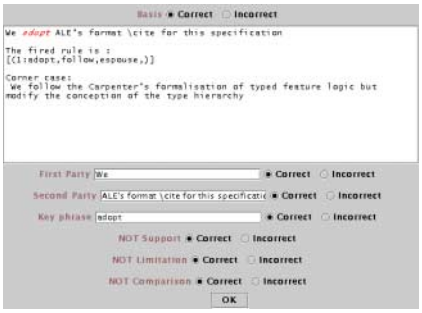
Within a group of the same ”context”, synonyms and hypernyms are also highlighted differently. In the event that the suggested lists and sublists of synonyms are not suitable for forming patterns, the user can also form their own word groups. These word groups can also be effectively reused by browsing quickly through existing word groups which are presented together with the synset-based word groups taken from WordNet.

Giving the user full control, as we do, does not appear to put an unreasonable burden on the user. By doing that, we also eliminate all potential problems which might come with an automatic preselection based on, e.g. a Part-of-Speech tagger. With the provided interface the user can easily select those words from one of the lists he or she deems appropriate. This is usually accomplished within 10s of seconds.

3.5 Templates

To build systems for more sophisticated tasks than simple sentence classification, KAFTAN allows users to develop concurrently for each category an information extraction knowledge base that fills the fields of a template with the respective information provided in the sentence. For each category the template can have different fields. What part of a sentence is filled into which slot is determined by a separate RDR-style knowledge base. For each slot in each template a separate RDR-tree is built. The knowledge for filling the templates is also incrementally acquired. Each time a sentence is correctly classified the user can check the content of the fields in the corresponding template. If a slot is incorrectly filled, the user will enter an exception rule to the rule responsible for the mistake. This gives us flexibility to refine any previously entered rule for both the classification task and the information extraction task. The conditions KAFTAN allows for the template extraction RDR node are the same as the condition in the classification RDR node.

The concept of templates for each class allows many different applications. In the case of citation classification which we discuss in this paper, it can be used to address the following problem: Sometimes it is impossible to tell what category a sentence belongs to just by looking at that sentence. This may for example be due to the fact that the authors introduce the cited work in one (or more) sentence(s) and describe the relationship later (in a sentence that does not contain any citations). In this situation, we can ”inspect” sentences around the citations and use templates to check if they talk about the citation.



Basis [✓ Correct] Incorrect

```
We adopt ALE's format \cite for this specification

The fired rule is :
[(1:adopt,follow,espouse,)]

Corner case:
 We follow the Carpenter's formalisation of typed feature logic but
modify the conception of the type hierarchy
```

First Party	We	[✓ Correct] Incorrect
Second Party	ALE's format \cite for this specificatic	[✓ Correct] Incorrect
Key phrase	adopt	[✓ Correct] Incorrect
NOT Support		[✓ Correct] Incorrect
NOT Limitation		[✓ Correct] Incorrect
NOT Comparison		[✓ Correct] Incorrect

Fig. 2. A screen shot of a KAFTAN window allowing users to provide the correct classification of a sentence. If a sentence is classified into a particular type, the fired rule and its cornerstone case are shown. The extracted phrases of all the slots in the corresponding template are also displayed. For each classification result that the user deems incorrect, s/he can tick the corresponding *Incorrect* box to enter a new exception rule

For example:

Taylor describes the Naive Bayes Classifier in \cite. In this paper, we adopt the Naive Bayes Classifier to do our classification task.

We cannot tell which category the first sentence belongs to. However the second sentence can be classified as *Basis* by the following rule:

(We|I|he|she) (adopt)

and the extracted fields for the template are:

- First party: *In this paper, we*
- Second party: *the Naive Bayes Classifier to do our classification task*
- Phrase: *adopt*

Using simple word matching, we can see that *Naive Bayes Classifier* mentioned in the first sentence appears in the *Second party* field which would suggest that it is a basis of the work in the *First party* field.

3.6 Building Citation Classifiers

The user is presented with the text (paragraph). He can either select a sentence with citation manually by highlighting the sentence with the mouse or ask the system to pick one automatically.

The system automatically classifies the selected sentence and show the result to the user. The result includes the classifications as well as the extracted fields for every category into which the sentence was classified (Figure 2). The user then verifies the displayed result. If an item was incorrect, the fired rule is presented so the user can see why the misclassification or incorrect information extraction has occurred.

To help the user decide how to fix a misclassification, the phrases that match the condition of the firing rule in the current sentence are highlighted. Together with the cornerstone case of the fired rule, it is fairly easy to come up with a new exception rule that differentiates the two.

The new rule is then checked against the existing rule base for consistency. There is a conflict if the new rule misclassifies a sentence that has been correctly classified before. In the conflict case, the new rule is rejected and the user has to construct a different condition possibly by modifying the previous one. The conflicting sentence would be presented to the user to help revising the condition.

The user has to go through this process until the new rule is accepted into the rule base. The sentence then is added into the list of previously seen cases for future consistency checks.

4 Experiments with KAFTAN

In this paper, we used four citation types, namely *Basis*, *Support*, *Limitation* and *Comparison*. In [9] 15 different citation types were distinguished. Our citation types appear to be the most relevant among those 15 types for the purpose of a citation map that is designed to support obtaining an overview of a field of research.

- *Basis*: One work is based on another work.
- *Support*: One work is supported by another work.
- *Limitation*: One work has been criticized to have some limits or weaknesses.
- *Comparison*: Two approaches are compared.

These citation types arguably reflect the most important relationships between papers that users are interested to know. They had been chosen before we acquired the corpus used in our experiment.

For each of these four categories, we have developed one SCRDR tree. We classify sentences and deem the citations occurring in those sentences to belong to the category of the sentence. Since a sentence could belong to more than one category we developed one RDR tree for each category so that one sentence may be classified into multiple categories.

Experiments In our experiments, we divided the data into two groups. The first group has 482 sentences while the second has 150 sentences. Initially, we used sentences in the first group to incrementally build our knowledge base(KB) using KAFTAN. As the result, our KB has 70 rules for the *Basis* class, 24 rules for the *Support* class, 23 rules for the *Limitation* class and 54 rules for the *Comparison* class. Each rule took about 2-3 minutes to be created by the user. The created KB is then expanded using 150 sentences from the second data group. At the end of this phase, the rule base ended up having 79 rules for the *Basis* class, 28 rules for the *Support* class, 30 rules for *Limitation* class and 59 rules for the *Comparison* class.

Table 1. Nanba & Okumura's results

		reference type identified by rules			accuracy for each type(%)
		C	B	O	
correct reference type	C	12	0	4	75.0
	B	2	25	5	78.1
	O	1	5	46	88.5

Every additional rule added was a result of one misclassification. Therefore the number of rules added reveals the number of error the rule base makes on second data group. To see how well KAFTAN performs, we compare our results against Nanba and Okumura's results which used the same corpus.

Nanba & Okumura used 282 reference areas for making rules and 100 for evaluation. A reference area in their approach usually contains multiple sentences. Their results are shown in Table 1.

Because Nanba & Okumura allow single classification and they have only two categories (excluding *Other*) we compare their accuracy against ours in classifying citations into a single class. Our *Limitation* and *Comparison* roughly corresponds to their TYPE C class whereas the BASIS category corresponds to the TYPE B class. Our *Support* category has no correspondence in their approach.

To make Nanba & Okumura's numbers comparable to our results we perform the following calculation which actually increases their accuracy numbers: When classifying citations into TYPE C, their system misclassified 4 TYPE C cases as seen in Table 1. Additionally, 2 TYPE B cases and 1 TYPE O case were classified incorrectly as TYPE C. This resulted in 7 misclassified cases out of 100 cases which is equivalent to an accuracy of 93%. Similarly, 7 cases of TYPE B were misclassified and 5 cases not of TYPE B were misclassified into TYPE B. Therefore the accuracy for TYPE B is (100-12) = 88%.

The accuracy of KAFTAN in classifying sentences into one category is calculated as follows. For the BASIS category, 9 rules were added which is equivalent to making 9 errors out of 150 cases. This results in an accuracy of (150-9)/150 = 94%. KAFTAN performed favourably in comparison to Nanba & Okumura's system (see Table 2).

Inspecting our knowledge base reveals that there are 47 exception rules not counting exception rules of the default rule. Every exception rule is the result of a misclassification of an existing rule. In approaches using a list of linguistic patterns, the rule that caused the misclassification needs to be modified. This might cause conflict as the modified rule usually has already correctly classified more than one sentence. This proves the usefulness of the rules structure in KAFTAN over the "flat" list of linguistic patterns.

Table 2. Our results using KAFTAN compared against Nanba and Okumura's results on the same set of documents

	# of rules with 482 sentences	# of errors in 150 citations	Accuracy
BASIS	70	9	94.0%
SUPPORT	24	4	97.3%
LIMITATION	23	7	95.3%
COMPARISON	54	5	96.7%
Nanba and Okumura's results			
TYPE B			88%
TYPE C			93%

5 Conclusion

This paper presented our system KAFTAN (Knowledge Acquisition Framework for Tasks in Natural language). KAFTAN allows users to rapidly develop knowledge bases for classification of sentences along with extracting information relevant for the particular sentence class.

Our results are very encouraging as the achieved classification accuracy is higher than a previous approach from the literature and the effort required with KAFTAN seems less than that in the comparison case in [7]. It should also be noted that while our measured accuracy is based on 'training data', the data was independent from the data used to build the bulk of the knowledge base. The accuracy measurements are taken from the last batch of 'training data' which did not result in a significant change of the knowledge base, i.e. if the knowledge base had not been changed on that last batch of data the same accuracy would have been measured. Hence our measurement is as good as measuring the accuracy on independent test data.

Further research will investigate the feasibility to automatically propose a list of possible conditions for discriminating the given sentence from cornerstone sentences. This would make the task of entering rules for the user even easier and we hope that the rules could become more general, so that the developed knowledge base can reach high accuracy even sooner.

Sentence classification was an essential subtask for our citation classification. But sentence classification is a recurring theme in other advanced tasks as well, including text summarization and information extraction.

We believe that the approach we took with KAFTAN is widely applicable for building powerful natural language processing tools. It represents a serious alternative to supervised machine learning approaches for NLP. While supervised machine learning approaches require a sizeable set of training instances which need to be classified, usually manually, we strongly believe that we can achieve the same or better results with significantly less data using the advice of a human who provides crucial information, e.g. in form of cue phrases, etc. to the system. This results in utilizing the time of the human in a more direct way to building the system. In fact, it looks like a detour to ask the human to classify training instances manually so that a program can find rules, instead of asking the human for the rules directly. While we haven't conducted rigorous studies in the NLP field as yet to confirm this point, studies in the medical domain involving fixed-length attribute vectors to be classified have been conducted and strongly support our point [5]. In NLP we expect this effect to be even more pronounced as the set of potential attributes to be used in conditions is almost infinitely larger than in the studies mentioned and hence, will hamper the automatic learning substantially, as is known from theoretical studies, such as the VC dimension, as well as practical studies.

Acknowledgements

We thank Hidetsugu Nanba for providing us with the corpus of scientific papers from the E-print archive which we used in our experiments and comparisons with Nanba & Okumura's work.

References

[1] J. Burstein, D. Marcu, S. Andreyev, and M. Chodorow. Towards automatic classification of discourse elements in essays. In *Proceedings of ACL-2001*, Toulouse, France, 2001. 762

[2] H. Edmundson. New methods in automatic extracting. *The Association for Computing Machinery*, 16(2):264–285, 1969. 762

[3] G. Edwards, P. Compton, R. Malor, A. Srinivasan, and L. Lazarus. PEIRS: a pathologist maintained expert system for the interpretation of chemical pathology reports. *Pathology*, 25:27–34, 1993. 760

[4] C. Fellbaum, editor. *WordNet - An electronic lexical database.* MIT PRESS, CAMBRIDGE, MA, 1998. 764, 765

[5] B. Kang, P. Compton, and P. Preston. Multiple classification ripple down rules: Evaluation and possibilities. In *Proceedings of the 9th AAAI-sponsored Banff Knowledge Acquisition for Knowledge Based Systems Workshop*, pages 17.1–17.20, 1995. 760, 771

[6] S. Lawrence, C. L. Giles, and K. Bollacker. Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71, 1999. 760

[7] H. Nanba and M. Okumura. Towards multi-paper summarization using reference information. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, 1999. 762, 770

[8] S. Tuefel. *Argumentative Zoning: Information Extraction from Scientific Text.* PhD thesis, Edinburgh, 1999. 762, 763

[9] N. Weinstock. Citation indexes. In A. Kent, editor, *Encyclopedia of Library and Information Science*, volume 5, pages 16–41. Marcel Dekker, New York, 1971. 768