Strojniški vestnik - Journal of Mechanical Engineering 53(2007)7-8, 429-441
UDK - UDC 025.4.03
Izvirni znanstveni članek - Original scientific paper (1.01)

Boljše izluščevanje informacij s pomočjo metode verjetnosti

Improving Information Extraction Using a Probability-Based Approach

Sanghee Kim[1] - Saeema Ahmed[2] - Ken Wallace[1]
([1]University of Cambridge, United Kingdom; [2]Technical University of Denmark)

Informacije imajo ključen pomen v celotnem trajanju izdelka. Rezultati raziskav kažejo, da se inženirji pogosto posvetujejo med seboj, da pridobijo informacije, potrebne za reševanje težav. Zaradi prehajanja ključnih kadrov v druga podjetja in upokojitev postaja vse bolj pomembna zmožnost izluščiti manjkajoče informacije iz tehnične dokumentacije, seveda, če le-ta obstaja. Zanimanje za različne načine izluščevanja tovrstnih informacij tako postaja vse večje. Iskanje po ključnih besedah je ustaljen način, vendar raziskave kažejo, da je ta način mnogokrat nezadovoljiv. Iskanje je mogoče izboljšati s standardiziranim načinom poimenovanja elementov in njihovih povezav v posameznih domenah, vendar je zaradi obilice sedanje dokumentacije, ki uporablja različna poimenovanja, izkoristek iskanja slab. Tako uporaba učenja, ki temelji na verjetnostnem modelu, obeta boljši izkoristek iskanja informacij ob enaki natančnosti, kakor ga omogočajo vnaprej določena iskalna pravila. Prispevek predstavlja rezultate iskanja informacij, ki temeljijo na verjetnostnem postopku. Preizkusi kažejo, da opisan postopek poveča izkoristek iskanja informacij s 53 odstotkov na 80 odstotkov, ob primerljivi natančnosti.

(Ključne besede: izluščevanje informacij, identifikacija dokumentov, iskalne strategije, klasifikacija, verjetnostne metode)

Information plays a crucial role during the entire life-cycle of a product. It has been shown that engineers frequently consult colleagues to obtain the information they require to solve problems. However, the industrial world is now more transient and key personnel move to other companies or retire. It is becoming essential to retrieve vital information from archived product documents, if it is available. There is, therefore, great interest in ways of extracting relevant and sharable information from documents. A keyword-based search is commonly used, but studies have shown that these searches often prove unsuccessful. Searches can be improved if domain entities of interest, e.g., 'gas turbine', are explicitly associated with their types, i.e., gas turbine is a type of engine, thus reducing the ambiguity of referring to the entities using various different ways of expressing them. It would be helpful to compile a full list of entities associated with the relevant types before identifying them in texts. However, due to the various ways of referring entities in the texts, manually defined identification rules tend to produce high precision but with low recall. In order to increase the recall, while maintaining the high precision, a learning approach that makes identification decisions based on a probability model, rather than simply looking up the presence of the pre-defined variations, looks promising. This paper presents the results of developing such a probability-based entity-identification approach. Tests show that the proposed approach achieves improved recall, i.e., from 53% to 80%, with comparable precision.

(Keywords: information searches, name entity identification, natural language processing, taxonomy, probability methods)

0 INTRODUCTION

Engineers frequently seek the information they need from colleagues. However, the time spent on acquiring and providing such information detracts from the time available to carry out their main tasks. In addition, as these engineers retire or move to other jobs, they can no longer be consulted. Engineers have to rely increasingly on documents, which are the prevalent information resource in organizations. Approximately 90% of organizational memory exists in the form of text-

based documents [1]. A computer-based document-management system can improve access to information contained in documents. Therefore, organisations are placing great emphasis on identifying reusable information in documents in order to improve access.

It has been reported that 35% of users find it difficult to access information contained in documents and at least 60% of the information that is critical to these organizations is not accessible using typical search tools [2]. There are three reasons for this problem. Firstly, there is simply too much information. Secondly, the keywords used by those searching for information might not be the same as those used to index the information. Thirdly, due to the nature of unstructured texts, it is a challenge to automatically index the information.

To address this problem, there have been many attempts to convert unstructured texts into more accessible formats. Recent research suggests the use of corporate taxonomies. As a hierarchical classification of entities, e.g., engineering products, a taxonomy supports the indexing and retrieval of information by annotating the content of a document with taxonomy entities, and then mapping a user's query onto those entities. In this way it is feasible to share and refer to the entities with less ambiguity. However, manually annotating texts with taxonomy entities can lead to time-consuming and error-prone indexing. Automatic annotating is thus preferred and this is one of the goals of Information Extraction (IE), which is a sub-field of Natural Language Processing (NLP). IE aims to extract entities that have pre-defined types, i.e., Named Entities (NE), using shallow NLP techniques.

The application of IE to engineering design document needs to pay particular attention to the difficulty of identifying engineering NEs, i.e., product names, due to their compositional features. This contrasts with typical IE systems that focus on NEs, which are usually identified using lexicon-syntactic grammar rules, i.e., people's names, dates, times. No previous research on developing IE systems for engineering design documents has been identified.

This paper presents a probability-based approach that achieves a good balance between precision and recall by automatically identifying NEs in engineering documents. Such an approach improves on the performance of the NE identification rules that simply enumerate pre-fixed variations and identify variations. A software prototype has been developed to test the performance of the proposed approach.

The overall aim of our research group is to understand how to make more information available in a readily useable form to engineering designers. The specific aim of the research described in this paper is to improve the IE from documents using a probability-based approach that is able to identify entities whose variations are not pre-defined.

1 LITERATURE REVIEW

Observations of engineering designers found that approximately 24% of a designer's time is spent acquiring and providing information [3]. This study also reported that in the aerospace industry, approximately 40,000 documents are produced during the design of a single aero-engine. Time could be saved if designers were able to retrieve the information they need from these documents rather than asking colleagues. However, this is only possible if designers are aware that relevant documents exist and they can retrieve the information contained in them easily. A further problem was identified by an empirical study that analysed 633 queries directed by novice designers to more experienced designers, which found that the novice designers were aware of what they needed to know in only 35% of all queries, i.e., they asked a specific question to which they received a specific reply [4]. These findings suggest that novice designers require support in identifying what they need to know. Such support could be provided through an explicit indexing structure that prompts the relevant queries.

Two common approaches to support information sharing in organizations are *knowledge repository* and *collaborative filtering*. A knowledge repository is a centralized resource where knowledge is structured into easily accessible formats [5]. The repository aims to explicitly represent knowledge for better distribution. One example is an expert system, which attempts to emulate the problem-solving ability of a domain expert by generating automatic solutions for a specific task. Expert systems have been widely deployed and have led to some significant improvements in knowledge sharing among employees. However, experience of using expert systems highlights the fact that organizational

knowledge does not remain static, so a dynamic knowledge-acquisition process is needed [6].

The second approach is to use collaborative filtering, which relies on the interactions between users to identify common task experiences and to recommend useful information [7]. It is easier to reuse information from the knowledge repository, but effort is required from the users to enter the information. Sharing through collaborative filtering does not impose such a burden on the users, but they do need to have a greater understanding of the context in order to retrieve the information they require. Whereas both approaches focus on capturing information, IE is more concerned with the indexing and retrieval of information. An empirical study by Furnas et al. [8] confirmed the importance of explicitly associating entities with types for retrieving needed information. They revealed that two users choose the same keywords for a single entity less than 20% of the time. In addition, according to industry experience, four synonyms are required for each entity [9].

Traditional IE systems use pre-defined templates that specify which NEs are to be extracted ([10] and [11]). However, templates can be too rigid to accommodate new types of entities and the relationships between them, so ontology or taxonomy-based IE systems are becoming increasingly popular ([12] to [14]). Ontologies and taxonomies make possible an inference-based approach that IE systems can use when the level of ambiguity is high.

One of the barriers to developing IE systems is their reliance on language experts for creating dictionaries of NEs along with extraction rules. The exploitation of learning techniques to reduce the reliance on human experts has therefore become popular. One such learning technique is supervised learning, which derives rules from manually tagged examples, e.g., 'AutoSlog' [15] and 'LTG' [16]. In AutoSlog, each extraction rule is defined by a trigger, a condition and a constraint in order to reduce the level of ambiguity. For example, the following rule identifies 'aircraft engine' as the entity of the 'started' event in the sentence: 'The aircraft engine was started by the pilot':

Trigger: verb should be 'started'
Condition: voice of sentence should be 'passive'
Constraint: subject should be a type of physical entity

The constraints are generally based on the syntax, implying that if the entities appear in different syntaxes, i.e., either as a subject or as an object, then two distinct rules need to be created to cover both cases [17]. LTG first identifies those text fragments that can be determined easily and delays the identification of the remainder until more evidence has emerged. This evidence includes 'position in a sentence' and 'whether the fragment is in lower case in a text'. It then uses a machine-learning method to identify the remaining text fragments.

The approach described in this paper incrementally creates the extraction rules from a limited number of examples of NE variations. In addition, the approach exploits the hierarchical classifications of NEs in order to recognize the more complex ones.

2 UNDERLYING CONCEPTS

2.1 Example

The following three unlinked extracts from Rolls-Royce's website[1] are used as examples throughout this paper.

<Extract 1> Like the motor car engine, the gas turbine is an internal combustion engine. In both, air is compressed, fuel added, the mixture ignited, and the rapid expansion of the resultant hot gas produces the power. However, combustion in a motor car engine is intermittent and the expanding gas produces shaft power through a piston and crank, whereas in a jet engine combustion is continuous and its power results from expanding gas being forced out of the rear of the engine.
<Extract 2> The intermediate case is a fabricated, spoked structure housing the thrust bearings for all shafts, and forming the air path between the IP and HP compressors. Externally it carries the A-frame support arms which brace the fan case (Module 7), and its internal hollow struts provide access for services such as oil tubes, cooling air, and the radial drive-shaft to the accessory gearbox.
<Extract 3> Largest of the modules, this is an assembly of forward and rear cylindrical casings and the fan outlet guide vane (OGV) ring. It is often referred to as the fan case. The titanium rear casing carries the fancase-mounted accessories and also contains acoustic linings. At their inner ends, the

[1] http://www.rolls-royce.com/education/schools/how_things_work/gasturbine/gasturbines.pdf

fan OGVs are secured to the torsion ring which locates the IP compressor module, whilst the outer ends are bolted to the front mounting ring. This assembly is welded to the titanium rear casing and bolted to the front casing.

Finding information using a keyword-based search is not always successful. For example, it may not be easy to find answers to the question: 'What material is used for the fan rear case'. For example, Google returns approximately 325,000 hits when it searches using the keywords 'fan rear case'. The first-ranked return is about improving the cooling system in a PC by adding a 'rear fan case'. Extract 3 above contains the answer to the query and it is available online. However, Google is unable to identify Extract 3 as it does not recognise that 'fan rear case' is the name of a product and that one of its variations is 'rear casing'. The search could be improved if Extract 3 had been indexed with NEs that identify product names and materials. If this had been done, Titanium could then be identified as the answer. Figure 1 shows the identification of the relevant NEs.

2.2 IE and Named Entity Identification

Using shallow NLP techniques, e.g., Part-Of-Speech (POS), IE can process a large number of documents effectively. It has demonstrated a significant improvement in retrieving relevant information compared to keyword searches.

IE structures information into easily accessible formats by identifying NEs and the relationships between them. NEs are pre-defined lists of domain entities. Generally, NEs are the proper names of organizations, people and locations. However, in this paper, NEs are text fragments referring to product names. For example, in the sentence 'Like the motor car engine, the gas turbine is an internal combustion engine.', 'turbine', 'gas turbine', 'engine' and 'car engine' could all be defined as product names. Highlighted texts, such as the one shown in Figure 1, help users to visually scan a large number of documents.

The three major tasks of IE are *NE identification, co-reference resolution and scenario filling*. NE identification involves the recognition of defined NEs and their variations. Co-reference is the referent shared by different entities. Its scope is broad, ranging from people (e.g., Joan and she) to objects (e.g., engine and it). Scenario filling integrates the extracted individual NEs into stories or new facts. This paper only addresses NE identification.

NE identification usually relies on two resources: (1) a dictionary of pre-defined NEs; and (2) extraction rules. The extraction rules are based on linguistic grammars specifying conditions under which the NEs are identified. For example, the following rule:

<Product> *is an assembly of* <Product>

states that if the left entity is identified as a type of 'Product', and then the right entity should also be tagged as 'Product'.

IE systems adopt different approaches. Systems with a complete list of NEs are less dependent on extraction rules, whereas systems with pattern-learning techniques need fewer pre-defined NEs. When an application domain generates new NEs frequently, it is difficult to manually maintain the NE dictionary, so a machine-

Largest of the modules, this is an assembly of <Product NE type=Front Casing>forward</Product> and <Product NE type=Fan Rear Case>rear cylindrical casings</Product> and the <Product NE type= Outlet Guide Van Inner Ring>fan outlet guide vane (OGV) ring</Product>. It is often referred to as the <Product NE type=Fan Containment Casing>fan-case</Product>. The <Material Type= Titanium>titanium</Material> <Product NE type=Fan Rear Case> rear casing</Product> carries the <Product NE type=Fan Containment Casing>fancase</Product>-mounted accessories and also contains acoustic linings. At their inner ends, the <Product NE type= Outlet Guide Van Inner Ring>fan OGVs</Product> are secured to the torsion ring which locates the <Product NE type= IP Compressor>IP compressor</Product> module, whilst the outer ends are bolted to the front mounting ring. This assembly is welded to the <Material Type= Titanium>titanium</Material> <Product NE type=Fan Rear Case> rear casing</Product> and bolted to the <Product NE type=Front Casing>front casing</Product>.

Fig. 1. *Example of NE identification*

learning technique is required. However, it is important to take into account the potential risk of relying entirely on the dictionary due to the possibility of polysemy. Polysemy is when an entity can be used in different contexts to express two or more meanings. For example, the entity 'bearing' has multiple meanings, e.g., (1) the 'bearing' supporting a rotating shaft and (2) the 'bearing' a ship is sailing on.

In engineering domains, NEs focus mostly on product and material names. The names of products and their components are relatively fixed, but have a large number of variations. For example, 'Fan Outlet Guide Vane Ring', can be shortened to 'Fan OGV Ring', and if the Ring is only used for the 'Fan' component, then the 'Fan' can be dropped, i.e. OGV Ring. In order to reflect this compositional structure, it is convenient to organise the product names in a hierarchical structure.

2.3 Taxonomy – NE Dictionary

The taxonomy employed as part of this research is Engineering Design Integrated Taxonomy (EDIT) [18]. One of the motivations for developing EDIT was to provide a visible indexing structure to help users search for information. There are two main advantages in having such a structure. Firstly, it helps designers focus their queries by browsing or navigating using the index; secondly, it provides the opportunity for search engines to recognise the context of a query. However, even if a search engine was able to recognise the context of a query, its results could only be as good as the original query. EDIT was developed by conducting interviews in two aerospace companies and analysing the transcripts of designers describing their design processes. EDIT consists of four root concepts:

1. The *design process* itself, namely, the different tasks undertaken at each stage of the design process. For example, conceptual design, detail design, brainstorming.
2. The *function* that must be fulfilled by all or part of a particular component or assembly. For example, one of the main functions of a compressor in a gas turbine is to 'compress air'.
3. The *issue* that the designer must consider while carrying out a stage of the design process. For example, considering the unit cost or manufacturing requirements.
4. The *product* itself, namely, component, sub-assembly and assembly. For example, outlet guide vane ring, fan case.

In this paper, NE identification is centred on the 'Product' root concept of EDIT, which currently has 220 entities defined. The 'Product' NEs are defined by both the 'Part-of' relation, e.g., 'IP compressor' is a part of 'Engine'; and 'Type-of' relation, e.g., 'IP compressor' is a type of 'Compressor'. One of the authors examined the 'Product' root concept from the perspective of NE identification and observed that:

- The names of 'Product' NEs in EDIT were largely determined from interviews with practising engineers and were the names they tended to prefer. However, texts contain an even wider range of names and many variations.
- As sub-classes are not always extended with the names of super-classes in EDIT, it can be difficult to correctly identify all the names simply through the taxonomy.

3 A PROPOSED APPROACH TO NE IDENFITICATION

3.1 Analysis of Product NEs in a Text

It is important to base NE identification systems on solid empirical evidence. The datasets provided by the Message Understanding Conference (MUC) have commonly been used for evaluating new identification systems [19]. However, the datasets define only seven types of NEs, i.e., Organisation, Person, Location, Date, Time, Money, and Percent, and product names are not included. In addition, no hierarchy is used for defining the NEs. Therefore, it was necessary to collect a new dataset in order to evaluate our approach. To do this 137 sample documents were collected from the same company with which EDIT was developed. These are one-page problem reports that describe problems, suggestions or new requirements that arose during product development. Once a new report is filed, senior engineers determine what further action is required to solve the problem described. A list of acronyms with their full definitions was provided by the company. One of the authors manually read each report and divided it into paragraphs, each of which was separated into sentences. After this the text fragments that contained relevant NEs were

identified, e.g., 'Forward Casing'. Each NE was then compared with the associated NE in EDIT, i.e., 'Front Casing', in order to identify any differences and how these differences could be defined.

Table 1 shows the results of the comparisons, which are split into seven categories: (1) singular or plural, (2) acronym, (3) compounds, (4) syntactic variation, (5) synonym, (6) separate reusable suffix, and (7) separate reusable modifier. Identical matches were included in the singular/plural variation. A total of 54% of the text fragments in the sample documents were matched with EDIT's Product NEs using one or more of the seven variations in Table 1. These seven variations are referred to as Exact Rules in this paper.

The first and second variations were identified frequently, singular/plural 24% and Acronym 13%, respectively. For example, the fragment 'HP Compressors' is matched with EDIT's 'HP Compressor' NE by identifying the plural form of 'Compressor'. Acronyms, e.g., 'OGV' for 'Outlet Guide Vane' can easily be matched from a list of definitions.

Compounds, e.g., 'Fancase' for 'Fan Case', are easy to match and syntactic variations, e.g., 'Starter Duct' and 'Starter Ducting', can be matched using POS tagging. However, the identification of synonyms, e.g., 'Forward Casing' and 'Front Casing' is more difficult since it requires appropriate definitions of the meanings of words. Synonyms can be matched by querying the lexical database WordNet [20].

Single and multiple terms are used for the names of Product NEs and it is difficult to identify the variations of NEs with multiple terms. In general, a multiple-term NE consists of a headword, the categorical part that contains the basic meaning, and a modifier that restricts the meaning. In the example 'Fan Case', the headword, 'Case' defines that it is a case for the fan. It is common practice to place the headword as the last of the terms. A separate reusable suffix, e.g., 'System' in 'Casing Cooling System', is part of an NE that can be removed without changing the overall meaning. Separate reusable modifiers, e.g. 'Electrical' in 'Electrical Harness', are often omitted because their purpose is to emphasise the headword and their meaning can be inferred.

It is necessary to combine multiple variations in order to correctly identify some fragments. For example, the fragment 'Case Cooling' is matched with 'Casing Cooling System' by identifying that 'Case' and 'Casing' are synonyms and 'System' is a removable suffix.

First, the Exact Rules are applied to determine if a fragment matches only one of EDIT's NEs. If more than one NE is matched, then the matching is ambiguous and the fragment remains untagged.

The remaining 46% of the text fragments in the sample documents were not uniquely matched with any of EDIT's Product NEs using the Exact Rules. Therefore, matching was based on the engineering judgment of the author who manually identified the NEs in the sample documents. Some examples are shown in Table 2. These NEs are difficult to extract automatically and a probability-based approach is necessary. Such an approach is the focus of this paper and is referred to as the application of Inexact Rules.

Table 1. *Exact Rules*

Variations	Example		Occurrences (%)
	Text fragment	EDIT Product NE	
Singular/plural	HP Compressors	HP Compressor	24%
Acronym	OGV	Outlet Guide Vane	13%
Compounds	Rear Fancase	Rear Fan Case	5%
Syntactic variation	Starter Duct	Starter Ducting	4%
Synonym	Forward Casing	Front Casing	3%
Separate reusable suffix	Case Cooling	Casing Cooling System	3%
Separate reusable modifier	Harness	Electrical Harness	2%
		Total	54%

Table 2. *Examples of partial matching*

Text fragment	EDIT Product NEs
LP Compressor Titanium Containment Case	Fan Containment Casing
Combustor Outer Casings	Combustion Outer Case
Fan Case Containment	(1) Fan Containment Casing (2) Rear Fan Case

The first fragment in Table 2, 'LP Compressor Titanium Containment Case', is matched with EDIT's 'Fan Containment Casing' NE by recognising that the 'Fan Containment Casing' is a part of 'LP Compressor' according to the 'Product' root concept in EDIT.

The second fragment 'Combustor Outer Casings' is not easily matched with 'Combustion Outer Case' although 'Combustion' and 'Combustor' are defined as synonyms. This is because 'Combustor', which is a part of the fragment, is defined as a single NE in EDIT. In order to achieve a correct identification, longer-length fragments should be clustered and a match attempted before single-word fragments are matched.

The third fragment 'Fan Case Containment' is matched with EDIT's 'Fan Containment Casing' NE if the *number* of shared terms is considered. However, if the *sequence* of terms is considered to be more important, then EDIT's 'Rear Fan Case' NE would be identified.

When identifying the text fragments above, it was necessary to take into account hierarchical or compositional descriptions of the NEs in EDIT. NE identification, therefore, does not rely entirely on the information from the dictionary. Instead, it is essential to pay special attention to the fragments that are partially matched with multiple NEs in EDIT. In summary, the following three guidelines are used to determine which of EDIT's NEs should be selected for each of the examples above. Preference should be given to the NE:

1. whose hierarchical descriptions match
2. with a 'maximum-length' match
3. composing terms preserve the partial ordering.

3.2 A Probability-based NE Identification

The proposed approach uses both the Exact and Inexact Rules. The Exact Rules contain a total of 1160 variations of the 220 Product NEs in EDIT, i.e., approximately five variations for each NE.

The Inexact Rules are based on naïve Bayesian probability. They are based on a simplified theorem that assumes variables to be independent in each class. In order to take into account the compositional descriptions of the NEs in EDIT, each fragment is encoded with the following attributes: (1) the two adjacent words; (2) POS tags; (3) the NE assigned to the previous text fragment; (4) the headword; and (5) all possible partial orders of the composing words of the fragment, with their orders preserved. Some of these encodings are described in the literature [16] and [21].

Given a text fragment, e_i, represented as a set of attributes, $t = (t_1, t_2 \ldots t_n)$, extracted from a text, $P(c_j \mid e_i)$, that represents the probability that Product NE, c_j, will be the NE against which will be matched [22]. This probability is defined as:

$$P(c_j \mid e_i) = P(c_j) \prod_{n \in positions} P(t_n \mid c_j)$$

with:

$$P(t_n \mid c_j) = \frac{(k_n + 1)}{(N + |vocabulary|)}$$

where is the number of times the attribute occurs in the NE ; N is the total number of attributes in the NE ; *vocabulary* is the set of all the distinct attributes for all the NEs; |*vocabulary*| is the total number of distinct attributes for all the NEs; *positions* is the set of attributes that appear in both the text fragment and *vocabulary*; and with:

The naïve Bayesian approach compares the attributes of a new text fragment with those of every NE in EDIT and computes the probability for each. The NE highest probability is assumed to be the best match. In this research it is assumed that each text fragment is assigned exactly to one Product NE. is the conditional probability that a fragment belongs to a particular Product NE, given that the fragment has the attribute values . Since it is difficult to estimate precisely all the combinations of the

values of the attributes, the conditional probability of each attribute value, $P(t_n \mid c_j)$, is computed instead based on the 'independence' assumption. Specifically, this means that the occurrence of a particular value of a specific attribute is statistically independent of the occurrence of any other attribute when predicting the Product NE of the fragment. The final probability is the product of the probabilities of all the attribute values in $t = (t_1, t_2 \ldots t_n)$. This conditional probability, $P(c_j \mid e_i)$, called the posterior probability, is then used to predict the Product NE for the next text fragment.

3.3 Software Prototype

To demonstrate the proposed approach, a software prototype was developed. Figure 2 shows the overall architecture of the prototype. It consists of two components: (1) automatic identification; and (2) verification. The first component was programmed using the Perl programming language, and the second component was programmed in Java and Protégé API[2]. The Protégé API was used for loading and displaying the EDIT Product NEs, which are specified in RDFS format. Extract 3 in Section 3.1 is used as an example.

3.3.1 Automatic identification

Step 1: Text Processing

The Text Processing step analyses a text with the following shallow NLP techniques.

1.1 Pre-processing

One paragraph is identified in Extract 3, which is then decomposed into five sentences. The first sentence is:

Largest of the modules, this is an assembly of forward and rear cylindrical casings and the fan outlet guide vane (OGV) ring.

Terms are identified as words lying between two spaces including the full stop.

1.2 Syntactic parse

The Apple Pie Parser [23] is used for a syntactic parse that tags Parts of Speech (POS) and identifies phrases. The Apple Pie Parser refers to the grammars defined in the Penn Treebank to determine the POSs [24]. For example, the first word 'Largest' is tagged as JJS, i.e., a superlative adjective. The remaining POSs for the sentence above are shown below:

POS taggings: *Largest/JJS of/IN the/DT modules/NNS, this/DT is/VB an/DT assembly/NN of/IN forward/JJ and/CC rear/JJ cylindrical/JJ casings/NNS and/CC the/DT fan/NN outlet/NN guide/NN vane/NN (OGV)/NNPX ring/NN.*

Phrase identification groups words grammatically, e.g., into Noun Phrases (NPs) such as {Largest of the modules} and {an assembly of forward and rear cylindrical casings}.

1.3 Lexical look-up

Each POS-tagged word is compared with WordNet definitions to achieve term normalisation. Acronym identification extends an acronym found in a text fragment with its full definition. An example of term normalisation is:

modules → module, casings → casing

and of acronym identification is:

OGV → Outlet Guide Vane.

Step 2: NE Identification

The NE Identification step takes the text fragments processed by Step 1 as an input and applies the Exact and Inexact Rules in turn.

2.1 Exact Rules

If the text fragment is identified as having a pre-defined variation, then the NE is identified and the process for that fragment is complete.

An example of a Product NE identified by the Exact Rules is:

Forward Casings → Front Casing (NE).

2.2 Inexact Rules

A probabilistic matching is required for those fragments that are not covered by the variations and that have multiple occurrences in the variations.

An example of a product NE identified by the Inexact Rules is:

Fan outlet guide vane (OGV) ring
→ Outlet Guide Vane Inner Ring (NE)

[2] http://protege.stanford.edu/doc/pdk/kb-api.html

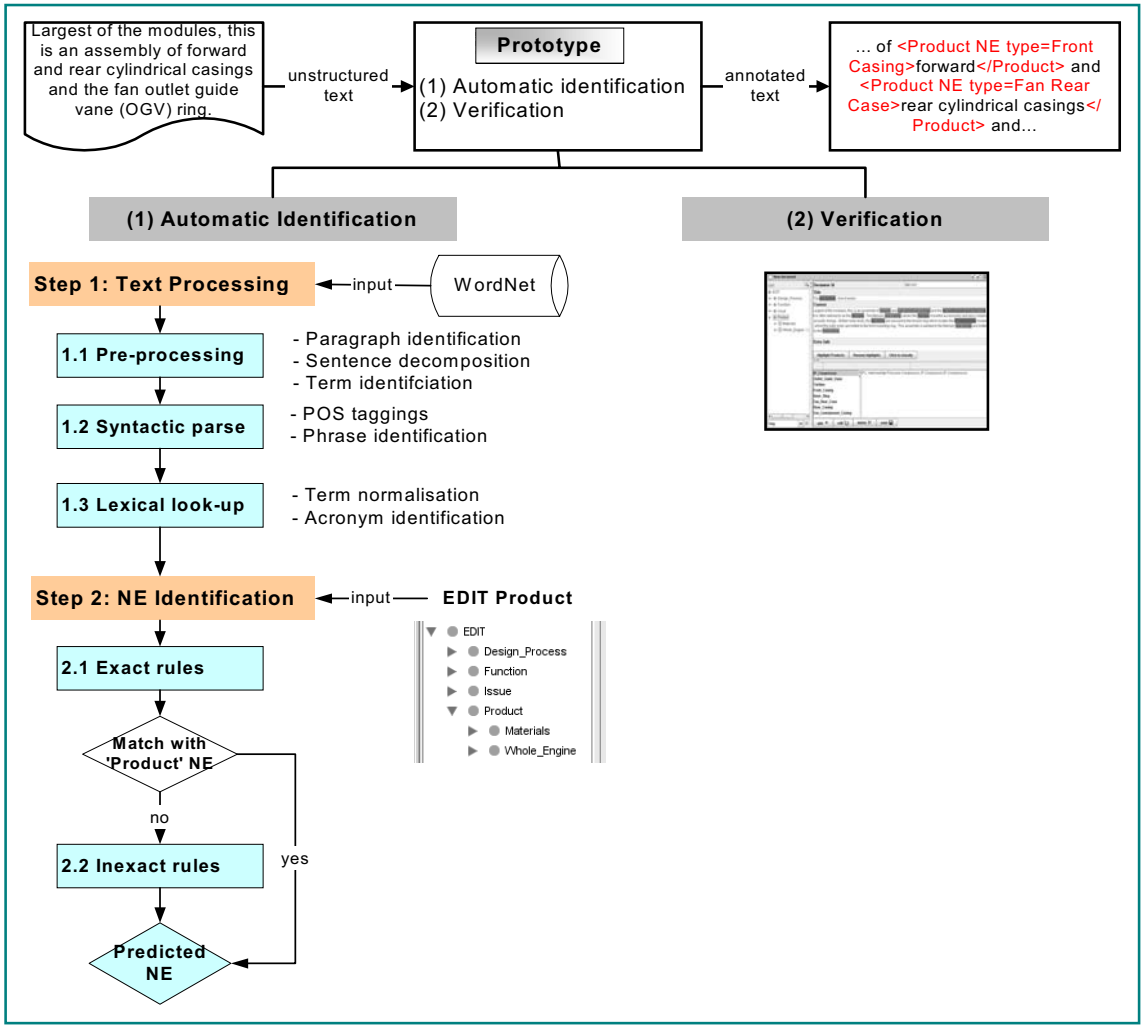

Fig. 2. *Architecture of the software prototype*

The full lists of identified Product NEs for Extract 3 given in Section 3.1 are shown in Figure 1 above.

3.3.2 Verification

A graphical interface has been developed for the prototype software with two specific objectives: (1) to assist the manual annotation process; and (2) to help visualize the results of automatic NE identification. Figure 3 is a screenshot of the interface showing Extract 3 as a sample text.

A user can click the 'Highlight Products' button, and all the text fragments that have been matched with Product NEs, shown as tree view in the left-hand frame, are highlighted in the text frame. Fragments that have the same NEs are shown in the same colour. A lower frame shows a list of the Product NEs that have been identified in the text. By clicking one of these NEs, its position in the tree view is highlighted. The corresponding variations are also shown in the adjacent frame. For NEs that have been incorrectly identified, the user can click the 'edit' button and make corrections. Similarly, missing NEs can be added by pressing the 'add' button. Each time the user presses the 'save' button, the text is identified as a new example and used by the naïve Bayesian approach for training.

4 TESTING THE PROTOTYPE

The prototype was tested to determine if using the probability-based Inexact Rules improved

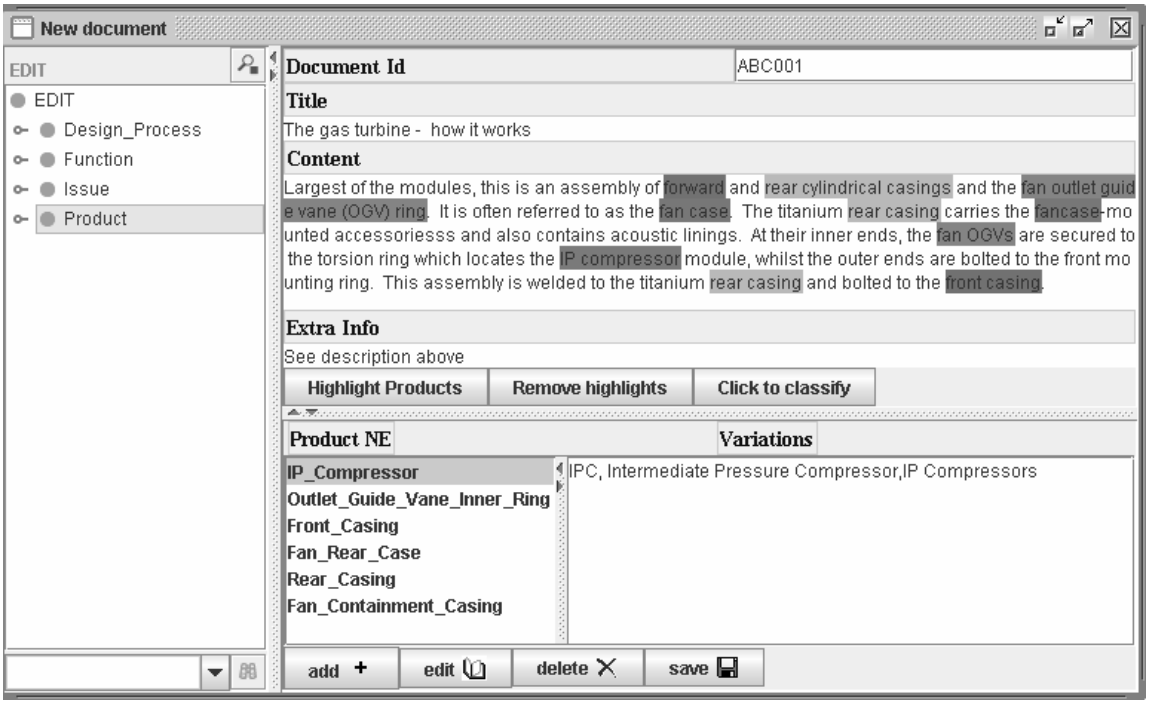

Fig. 3. *An example screenshot of the prototype*

the Product NE extraction over and above that achieved by the Exact Rules. A total of 267 problem reports, including the 137 reports used in Section 4.1, were used for the testing. A total of 977 text fragments were identified with 977 EDIT NEs. EDIT has 220 Product NEs and in the 267 reports 101 of them appeared, i.e., 119 NEs from EDIT did not appear in any of the reports.

From the 977 fragments and their assigned NEs, 50% were randomly selected for training the prototype and the remaining fragments were used for the testing. Precision and recall were used for measuring the prototype's performance. In this paper, recall is defined as the percentage of the test fragments that were matched with EDIT's NEs, whether these were correct or not. Precision is defined as the percentage of the recalled fragments that were correctly matched. Table 3 shows the results of the evaluation.

Overall, the prototype, using both the Exact and Inexact Rules, achieved a good balance of recall and precision, i.e., 80% recall and 81% precision. Using the Exact Rules on their own, the recall was 53% and the precision was 85%. The addition of the Inexact Rules therefore significantly increased the recall while maintaining the precision. The Exact Rules also failed to identify 20 of the 101 EDIT NEs that appeared in the reports. However, it is surprising that the precision achieved by the Exact Rules was not closer to 100%. This means that some of the Exact Rules are, in fact, not 'exact'. The reason is that the empirical evidence, from which the rules were derived, did not cover all the possible variations in EDIT and examples of ambiguity remained. For example, the Exact Rules included 'Stubshaft' as a variation of EDIT's 'Stub Shaft' NE, which is a part of 'Turbine'. However, the 'Stub Shaft' component is also a part of 'Compressor' and this variation was not included in the rules.

Ideally the Inexact Rules should be trained with the minimum number of examples. The effect of changing the number of fragments used for training is shown in Figure 4, where accuracy is defined by the number of correct NE identifications divided by the total number of NEs identified. The 977 fragments were divided into ten equal-sized clusters, each of which was used to estimate the accuracy at that point. For example, when tested with only 100 fragments, the accuracy was observed to be 42%. The maximum accuracy achieved was 84%. It was noticeable for this relatively small total sample size of 977 fragments that the accuracy was relatively constant at around 70% for between 400 and 800 training samples.

Table 3. *NE identification results*

NE Identification	Recall	Precision
Exact Rules only	53%	85%
Exact + Inexact Rules	80%	81%
Change	+27%	–4%

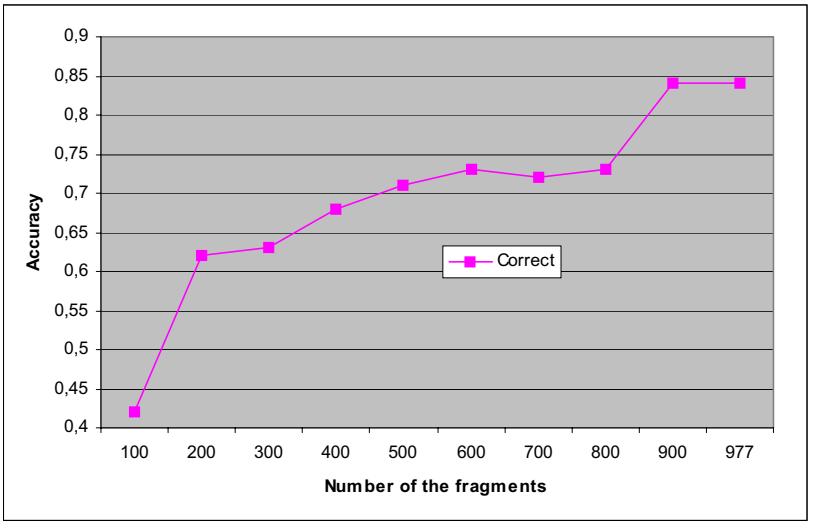

Fig. 4. *Naïve Bayesian predictions estimated for different numbers of NEs*

5 CONCLUSIONS AND FUTURE WORK

It is well established that when engineers need information they tend to ask colleagues in the first instance. However, the industrial world is now more transient and key personnel are retiring and frequently moving, either within their own organisations or to other organisations. It is becoming necessary to retrieve more information from archived documents. There is, therefore, great interest in ways of extracting such information. Information contained in a document is easier to retrieve if domain entities are explicitly tagged with their types so they can be more easily identified using keyword searches.

However, it is particularly difficult to automatically identify engineering entities, i.e., Product NEs, due to the hierarchical and compositional nature of their descriptions. Two IE approaches for achieving this are described in this paper: Exact Rules and Inexact Rules, the latter being based on naïve Bayesian probability. When using the Exact Rules on their own the recall was 53% and the precision was 85%. By adding the Inexact Rules the recall increased significantly to 80% and the precision remained good at 81%. State-of-the-art IE systems are known to demonstrate comparable recall and precision to the proposed approach. However, those systems have mainly been applied to limited types of NEs that do not use a hierarchy to classify them. This research has focused on a wider range of engineering Product NEs that use a hierarchical taxonomy to classify them.

The main contributions of this research are: (1) the observation that Product NEs are more complex than other types of NE; and (2) the development of a probability-based NE identification approach to identify correctly these complex NEs in texts. The improved recall achieved, i.e., from 53% to 80%, is believed to be due to the reliable and robust probability estimates based on how the various combinations of the attributes in a given fragment are related to the correct Product NE.

One of the shortcomings of using a supervised learning approach, e.g., naïve Bayesian probability, is that because it needs examples from which the probability distributions are derived, no predictions can be made for a given NE if that NE

has not appeared in any examples. In current practice, the identification is made incrementally, i.e., with each example the probability distribution is computed and revised, so that the existence of a new example is only recognised manually after the identification is evaluated incorrectly. Since the probability model might need a certain number of examples in order to learn a correct probability distribution, it might take some time to predict a Product NE correctly. This can be improved on by first clustering similar examples into groups without considering the potential NEs. In this way, it is easy to identify the list of Product NEs that might need further examples.

When the hierarchy of the *Product* root concept in EDIT changes, i.e., a new concept is added or an existing one is deleted, the existing probability model has to be revised. In doing so, it is necessary to check whether this change affects the existing model. Currently, the probability model is re-computed when such a change occurs. This can be improved by revising only the part of model where the change needs to be reflected.

Acknowledgements

This research was funded by the University Technology Partnership for Design, which is a collaboration between Rolls-Royce, BAE SYSTEMS and the Universities of Cambridge, Sheffield and Southampton.

6 REFERENCES

[1] Rosner, D., Grote, B., Hartman, K., and Hofling, B., (1998), From natural language documents to sharable product knowledge: a knowledge engineering approach, In Information Technology for Knowledge Management, 8, *Springer*, New York., pp.151-182.

[2] 80-20 software., (2003), 80-20 retriever enterprise edition, http://www.80-20.com/brochures/Personal Email Search Solution.pdf

[3] Marsh, J.R., and Wallace, K., (1997), Observations on the role of design experience, *WDK Annual Workshop*, Switzerland.

[4] Ahmed, S., and Wallace, K. M., (2004), Understanding the knowledge needs of novice designers in the aerospace industry, *Design Studies*, Vol. 25, No. 2, pp. 155-173.

[5] Donnellan, B., and Fitzgerald, B., (2003), A KMS application to support knowledge sharing in a design engineering community, *Proceedings of the Eleventh European Conference on Information Systems*, Italy.

[6] Otto, K., and Abecker, A., (1998), Corporate memories for knowledge management in industrial practice: prospects and challenges, In Information Technology for Knowledge Management, 8, *Springer*, New York, pp. 183-206.

[7] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J., (1999), An algorithmic framework for performing collaborative filtering, *Proceedings of the 22nd Conference on Research and Development in Information Retrieval*, U.S.A., pp.230-237.

[8] Furnas, G., Landauer, T., Gomez, L., and Dumais, T., (1986), Statistical semantics: analysis of the potential performance of keyword information systems, Bell System, *Technical Journal*, Vol. 62, No. 6, pp.1753-1806.

[9] Gilchrist, A, (2001), Taxonomies for business: knowledge representation, *Proceedings of the 11th Nordic Conference on Information and Documentation*, Iceland

[10] Collier, R., (1996), Automatic template creation for information extraction, an overview, PhD thesis, *University of Sheffield*.

[11] Grishman, R., (1997), Information extraction: techniques and challenges, In *Lecture Notes in Artificial Intelligence*, Vol. 1299.

[12] Kim, S., Lewis, P., Martinez, K., and Goodall, S., (2004), Question answering towards automatic augmentations of ontology instances, *Proceedings of the first European Semantic Web Symposium*, pp. 152-166.

[13] Maedche, A., Neumann, G., and Staab. S., (2002), Bootstrapping an ontology-based information extraction system, *Intelligent Exploration of the Web*, editor (J. Kacprzyk).

[14] Staab, S. Maedche, A., and Handschuh, S., (2003), Creating metadata for the semantic web, an annotation environment and the human factor, *Computer Networks*, Vol. 42, pp.579-598.
[15] Riloff, E., (1993), Automatically constructing a dictionary for information extraction tasks. *Proceedings of the 11th National Conference on Artificial Intelligenc*e (AAAI-93), pp. 811–816.
[16] Mikheev, A., Grover, C., and Moens, M., (1998), Description of the LTG system used for MUC-7, *Proceedings of the Seventh Message Understanding Conference*, Virginia.
[17] Muslea, I., (1999), Extraction patterns for information extraction tasks: a survey, *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction*, Florida.
[18] Ahmed, S., (2005), Encouraging reuse of design knowledge: a method to index knowledge, *Design Studies*, Vol. 26, No. 6, pp.565-592.
[19] Marsh, E., and Perzanowski, D., (1997), MUC-7 evaluation of IE technology: overview of results, *Proceedings of the Seventh Message Understanding Conference*.
[20] Miller, G.A., Beckwith, R.W., Fellbaum, C., Gross, D., and Miller, K., (1993), Introduction to wordnet: An on-line lexical database, *International Journal of Lexicography*, Vol. 3, No. 4, pp.235-312.
[21] Paliouras, G., Karkaletsis, V., Petasis, G., and Spyropoulos, (2004), Learning decision trees for named-entity recognition and classification, *Proceedings on the Workshop on Machine Learning for Information Extraction*, ECAI, Germany.
[22] Mitchell, T., (1997) Machine learning, McGraw Hill.
[23] Sekine, S., and Grishman, R., (2001), A corpus-based probabilistic grammar with only two non-terminals, *Proceedings of the 1st International Workshop on Multimedia annotation*.
[24] Macus, M.P., B. Santorini, M., and Marcinkiewicz, A. M.. (1994) Building a large annotated corpus of English: the penn treebank. *Computational Linguistics* 19 (2), 313-330.

Authors' Addresses:
Dr. Sanghee Kim
Prof. Dr. Ken Wallace
University of Cambridge
Department of Engineering
Trumpington Street
Cambridge, CB2 1PZ, U.K.
shk32@eng.cam.ac.uk
kmw@eng.cam.ac.uk

Prof. Dr. Saeema Ahmed
Technical University of Denmark
Department of Mechanical Engineering
Building 404
DK-2800 Lyngby, Denmark
sah@mek.dtu.dk

Prejeto: 2.5.2007
Received:
Sprejeto: 27.6.2007
Accepted:
Odprto za diskusijo: 1 leto
Open for discussion: 1 year