Assessing the Influence of Personal Preferences on the Choice of Vocabulary for Natural Language Generation

Raquel Hervás[a,*], Virginia Francisco[a], Pablo Gervás[b]

[a]*Departamento de Ingeniería del Sofwtare e Inteligencia Artificial, Universidad Complutense de Madrid, 28040, Madrid, Spain*
[b]*Instituto de Tecnología del Conocimiento, Universidad Complutense de Madrid, 28040, Madrid, Spain*

Abstract

Referring Expression Generation is the part of Natural Language Generation that decides how to refer to the entities appearing in an automatically generated text. Lexicalization is the part of this process which involves the choice of appropriate vocabulary or expressions to transform the conceptual content of a referring expression into the corresponding text in natural language. This problem presents an important challenge when we have enough knowledge to allow more than one alternative. In those cases, we need some heuristics to decide which alternatives are more appropriate in a given situation. Whereas most work on natural language generation has focused on a generic way of generating language, in this paper we explore personal preferences as a type of heuristic that has not been properly addressed. We empirically analyze the TUNA corpus, a corpus of referring expression lexicalizations, to investigate the influence of language preferences in how people lexicalize new referring expressions in different situations. We then present two corpus-based approaches to solve the problem of referring expression lexicalization, one that takes preferences into account and one that does not. The results show a decrease of 50% in the similarity error against the reference corpus when personal preferences are used to generate the final referring expression.

Keywords:

*Corresponding author

Email addresses: raquelhb@fdi.ucm.es (Raquel Hervás), virginia@fdi.ucm.es (Virginia Francisco), pgervas@sip.ucm.es (Pablo Gervás)

Natural Language Generation, Referring Expression Generation,
Lexicalization, Personalization, Corpus Approach

1. Introduction

Natural Language Generation (NLG) is a subfield of Artificial Intelligence and Computational Linguistics that covers the design and construction of systems that produce text in human languages. The general process of text generation (Reiter & Dale, 2000) takes place in several stages, during which non-linguistic input is progressively refined by adding information that will shape the final text. During the initial stages the concepts and messages that will appear in the final content are decided and these messages are organized into a specific order and structure (*content planning*), and particular ways of describing each concept where it appears in the discourse plan are selected (*referring expression generation*). This results in a version of the discourse plan where the contents, the structure of the discourse, and the level of detail of each concept are already fixed. The *lexicalization* stage that follows decides which specific words and phrases should be chosen to express the domain concepts and relations which appear in the messages. A final stage of *surface realization* assembles all the relevant pieces into linguistically and typographically correct text.

Referring Expression Generation (REG) is one of the most studied problems of the Natural Language Generation process. Many different solutions have been proposed for this task (Dale & Haddock, 1991; Dale & Reiter, 1995; Krahmer & Theune, 1998, 2002; Krahmer et al., 2003; Gatt, 2007), each one taking into account different considerations and approaches. Much work has been devoted to the selection of the conceptual content of a referring expression; in contrast, the choice of lexical items and syntactic structures for referring expressions has not been treated in depth.

Within the Referring Expression Generation process, Lexicalization chooses an appropriate word or phrase to express each conceptual part of the referring expression. For example, if the intended referent is a man, there are different choices such as *man*, *guy*, *gentleman*, etc., depending on factors such as context or style. The implementation of a referring expression generation algorithm with the capacity to perform lexical choices presents an important challenge. In those contexts where we have more than one alternative for the lexicalization of the elements we need heuristics to decide which alternatives

are more appropriate for each reference to an object in a given text. These heuristics should take into account aspects such as terminological restrictions or common practices according to different styles or situations.

Although different systems have tried to model the "correct" way of creating referring expressions, we believe that there is not a single "correct" way of performing this task (especially for lexicalization), as it greatly depends on the person, situation or channel in which the communication is being developed. As in other human-computer interaction tasks, such as query-based web searches (Lorigo et al., 2006) or tag query in social media (Clements et al., 2010), personalization can improve the final results when generating natural language aimed at a specific user or group of users. For example, it is possible to create different user models that can be used to configure generation modules depending on the type of user being addressed and how they use language themselves. This kind of personalization effort would be useful if the system must accommodate its responses to a specific genre or author style, or even align its utterances to those previously used by the user in applications such as dialogue systems.

As a first step towards demonstrating this idea and providing personalized solutions for this problem, we empirically analyze a corpus of lexicalizations of referring expressions and study the influence of language preferences in how people lexicalize new referring expressions in different situations. In order to do so we present a corpus-based approach to referring expression lexicalization that relies on case-based reasoning (CBR), a suitable paradigm in this case as it is based on reproducing previous solutions to a problem. In order to compare the influence of personal preferences we compare the results obtained after training the system in two different ways: taking language preferences into account or not considering them.

We obtain a 50% decrease in the similarity error obtained by our system when considering personal preferences against the solution that does not. We think this is an important basis for considering new approaches to REG involving this kind of phenomenon, making them more human-like in their responses, or biased to specific types of users if there are available corpora showing their preferences.

This work is structured as follows. Section 2 and 3 present an outline of the Referring Expression Generation field and the Case-Based Reasoning paradigm. Section 4 shows our CBR approach to Referring Expression Lexicalization and Section 5 the results obtained when training the system in two different ways: one that takes language preferences into account and one

that does not. Finally, Section 6 outlines some conclusions and future work.

2. Referring Expression Generation

Referring Expression Generation (REG) is concerned with how to produce a description of an entity that enables the hearer to identify that entity in a given context (Reiter & Dale, 2000). This description can be a pronoun (*he*, *it*, etc.), a proper noun (*John*, *The Caledonian Express*, etc.), or a nominal phrase (*the train*, *the man*, etc.), which can then be complemented with attributes or relationships (*the Aberdeen train* or *the train on platform 12*). In each case, it will be necessary to take into account semantic information about the entities we want to refer to.

Nominal phrases (usually in the form adjectives + noun) are one of the most common forms to express a reference to an entity. In these references, the noun will usually correspond to the type or class of the referent, and the adjectives will correspond to its values for specific attributes. There are two important steps to take into account in order to generate such referring expressions: to decide which set of attributes can distinguish the entity from any other distracting entities and to decide how that information will be expressed in the text.

If we consider the type of entity we want to mention as given[1], the first step when creating a referring expression is to decide which set of attributes applicable to the entity can distinguish it univocally from any other distracting entities. This process is called *Attribute Selection*. Then, once we have selected the information that will be included in the referring expression (attributes + type), it is necessary to decide how that information will be expressed in the text. This step will require selecting which syntactic and lexical structures are most suitable to translate the conceptual information into text. This is mostly the task for the *Lexicalization* stage.

The Incremental Algorithm by Dale & Reiter (1995) is probably the most studied solution for the selection of attributes. The authors describe a fast

[1] In some situations the default type of an entity may not be the most appropriate choice given the context of the discourse. In those cases in which additional information about the entities is available (such as a taxonomy or a hierarchy of concepts), it will be possible to identify the level in which the discourse is taking place and the level that should be used to create the referring expression. However, we are not considering this problem here. The reader can consult (Hervás & Gervás, 2008) for more details.

algorithm for generating referring expressions in the context of a natural language generation system. The algorithm they present is based on psycholinguistic evidence. As such, it provides an acceptable baseline for the basic operations and the performance expected from such an algorithm.

This algorithm relies on the following set of assumptions about the underlying knowledge base that must be used: (1) every entity is characterized in terms of attribute-value pairs; (2) every entity has as one of its attributes a type; and (3) the knowledge base may organize some attribute values as a subsumption hierarchy.

To construct a reference to a particular entity, the algorithm takes as input a symbol corresponding to the intended referent and a list of symbols corresponding to other entities in focus, known as the *contrast set*. The algorithm returns a list of attribute-value pairs that correspond to the semantic content of the referring expression to be realized. The algorithm operates by iterating over the list of available attributes, looking for one that is known to the user and rules out the largest number of elements of the contrast set that have not already been ruled out. Information about basic level values is used to give preference to some attribute over another when the other criteria give no clear choice.

2.1. Lexicalization of Referring Expressions

Lexicalization is understood as the process of deciding which specific words and phrases should be chosen to express the domain concepts and relations which appear in a message (Reiter & Dale, 2000). This task also includes the choice of other linguistic resources which convey meaning, like for example particular syntactic structures. The most common model of lexicalization is one where the lexicalization module converts an input graph whose primitives are domain concepts and relations into an output graph whose primitives are words and syntactic relations. This scheme can be valid for most applications where the domain is restricted enough in order that direct correspondence between the content and the words to express it is not a disadvantage. In general, thinking of more expressive and versatile generators, this model requires some improvement.

Cahill (1998) differentiates between "lexicalization" and "lexical choice". The first term is used to indicate a broader meaning of the conversion of something to lexical items, while the second is used in a narrower sense to mean deciding between lexical alternatives representing the same propositional content. The choice of syntactic structures is included in Cahill's

"lexicalization", and it is usually referred to as "syntactic choice".

A classic example of lexical choice for natural language generation systems is Michael Elhadad's PhD Thesis (Elhadad, 1992), which addresses this particular problem for a system employing the FUF surface realizer (Elhadad, 1993). Elhadad's solution operates by unifying a conceptual representation of the input with a grammar that encodes the set of linguistic choices to be made during realization. The conceptual representation of the input is given as a *functional description*, basically a set of attribute-value pairs where the values can be functional descriptions themselves, enabling recursive construction of complex structures. The grammar used is an extended version of a functional description, with the added peculiarity that it allows the inclusion of choice points. During unification, for each choice point in the grammar the system attempts to unify the input with each of the alternatives until a matching one is found. The system allows backtracking in case of failures.

Bangalore & Rambow (2000) maintained that choosing the best lexeme to realize a meaning in natural language generation is a hard task. They investigated different tree-based stochastic models for lexical choice that relied on a corpus. Edmonds & Hirst (2002) developed a computational model of lexical knowledge that can adequately account for near-synonymy, and deployed such a model in a computational process that could "choose the right word" in any situation of language production.

From the point of view of the lexicalization of references, Horacek (1997) stated the problem of how most of the referring expression generation algorithms do not take into account the linguistic realization of the conceptual information they choose. This unawareness summed up in two assumptions found in most of these approaches: (1) the instant availability of lexical descriptors for the information to be conveyed, and (2) the adequate expressibility of the chosen set of information in terms of lexical items. In summary, besides the goal of producing a distinguishing reference to the intended referent, there are also the secondary goals of expressing the chosen information in a natural way and applying a suitable processing strategy.

Siddharthan & Copestake (2004) described an algorithm for generating referring expressions in open domains strongly based on lexical information. The algorithm works at the level of words, not semantic labels, and measures the relatedness of adjectives (lexicalised attributes) using the lexical knowledge base WordNet rather than a semantic classification. Janarthanam & Lemon (2009) addressed the problem of referring expression lexicalization in spoken dialogue systems where the automated system needs to adapt its gen-

eration choices to the users' lexical knowledge. They presented a statistical learning algorithm that adapts its vocabulary to the one employed by the user. Stoyanchev & Stent (2009) also considered that, when generating referring expressions in interactive settings, lexical choices are quite dependent on the previous dialogue history. The tendency to align the vocabulary with that of the other participants in a dialogue influences the choice of conceptual and textual content of the utterances.

2.2. Personal Preference Considerations in Referring Expression Generation

Although it has been generally considered that there could be many correct lexicalizations for the same referring expression and that there are no clear means of deciding between them, the issue of studying how personal preferences can influence this decision has almost not been addressed in the literature. In the last years some works have dealt with these issues, most of them in the scope of the shared tasks for REG (Belz & Gatt, 2007; Gatt et al., 2008a, 2009).

In (Bohnet, 2008, 2009) different algorithms for attribute selection and lexicalization of referring expressions were presented, all of them based on the individual styles of each of the authors of the TUNA corpus. From the point of view of lexicalization, different models of vocabulary and syntactic expressions, containing information about preferences in the use of determiners or favorite words, were created for different people.

Di Fabbrizio et al. (2008) also tried to consider stylistic differences between speakers, along with the relation between semantic and word order information. For the attribute selection task they studied the corpus in order to determine what were the most frequent usages of attributes for each of the authors of the corpus. For the lexicalization task they considered the different situations in the corpus as templates, but in such a way that if a set of attributes not present in the corpus was to be lexicalized, the system failed because of lack of a suitable template.

2.3. Evaluation for Referring Expression Generation

As in other fields within Artificial Intelligence, it is important to assess the generation of referring expressions in order to know if it is adequate. Some efforts have been made in the scope of a series of shared tasks for REG (Belz & Gatt, 2007; Gatt et al., 2008a, 2009). These tasks involved a significant number of participants in each of them.

In order to evaluate the similarities between the language generated by humans and the language generated by a generation system, it is possible to rely on the use of a corpus. There is a strong tradition of corpus-based evaluation in the field of Natural Language Processing. However, there has also been much discussion about whether it is possible to consider existing corpora as the reference to compare the quality of automatic algorithms (Reiter & Sripada, 2002). Taking into account that these resources are usually generated with a great number of authors, it is usual to find so much variation in them that it becomes impossible to extract a "correct" way in which the studied task must be performed. In the case of the generation of referring expressions, the use of a corpus is complicated for similar reasons. However, our goal is to assess the influence of heterogeneity in the generation of referring expressions. In this case, a corpus created by several authors would be a very useful resource.

2.3.1. TUNA Corpus

One of the most important projects in the field of referring expression generation is the TUNA project (van Deemter et al., 2006; Gatt et al., 2007). Under the TUNA project a corpus of referring expressions in the form of nominal phrases was developed for visual entities in the domains of people and furniture. The corpus was obtained during an experiment in which subjects were asked to write textual descriptions for target entities in a situation where there were also six other entities called distractors. Each referring expression from the corpus is accompanied by the conceptual representation of the situation in which it was generated. The TUNA corpus has been used in this work as a case base for our CBR solution and as reference for the evaluation of the implemented algorithm. An exhaustive description of the corpus can be found in (van der Sluis et al., 2006; Gatt et al., 2008b).

The TUNA corpus contains 398 XML documents for the furniture domain and 340 for the people domain. Each data file consists of a single example of the corpus, i.e. a pair consisting of a single situation (the representation of entities and their attributes) and a referring expression that describes an entity in that situation (the target). The basic format of the instances of the corpus consists mainly of the following nodes:

- *DOMAIN*: Representation of entities in terms of their attributes.
- *STRING-DESCRIPTION*: The string describing the target referent in the domain.

- *ATTRIBUTE-SET*: The set of domain attributes included in the description. Attributes represent the characteristics of an entity using attribute-value pairs. The possible attributes and values for both domains are shown in Table 1. Empty cells represent attributes that are not used in the domain. X-DIMENSION and Y-DIMENSION correspond to the coordinates of the referent in a 5 (column) x 3 (row) matrix in which the objects were presented during the experiment.

- *DESCRIPTION*: The string in STRING-DESCRIPTION where the relevant substrings are annotated with attributes from the ATTRIBUTE-SET. The substrings corresponding to attributes could be just words (*big*) or phrases (*that is big*).

Figure 2 in Section 4.1 presents an example of one of these XML documents.

Attribute	**Possible values**	
	Furniture	**People**
TYPE	chair, sofa, desk, fan	person
ORIENTATION	front, back, left, right	front, left, right
X-DIMENSION (column number)	1; 2; 3; 4; 5	1; 2; 3; 4; 5
Y-DIMENSION (row number)	1; 2; 3	1; 2; 3
SIZE	large, small	
COLOUR	blue, red, green, grey	
AGE		young, old
HASBEARD		0 (`false`), 1 (`true`)
HAIR-COLOUR		dark, light, other
HASHAIR		0 (`false`), 1 (`true`)
HASGLASSES		0 (`false`), 1 (`true`)
HASSHIRT		0 (`false`), 1 (`true`)
HASTIE		0 (`false`), 1 (`true`)
HASSUIT		0 (`false`), 1 (`true`)

Table 1: Attributes and their values in the two TUNA domains

There were 57 authors per domain who were identifiable via a unique identification number (ID). This ID was then used to identify the examples in the corpus created by each person. More specifically, each author created seven examples for the furniture domain and six for the people domain.

Mismatches between these numbers and the total number of elements in the corpus correspond to examples that were incorrectly classified and were not used in the experiments.

2.3.2. Evaluation Metrics

The evaluation metrics we have used in this work are the usual ones in the literature for assessing the final output of NLG systems. Even when they were intended as metrics for machine translation and document summarization, they have been widely used as intrinsic evaluation metrics for comparison of system vs. human generated texts, specially in the REG Shared Tasks (Belz & Gatt, 2007; Gatt et al., 2008a, 2009) and later studies (Gatt & Belz, 2010):

- **String-edit distance (Levenshtein, 1966).** Also called Levenshtein distance, it is a metric for measuring the amount of difference between two sequences (in this case the sequences compared are the string generated by the system and the one in the corpus). The Levenshtein distance between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation is an insertion, deletion, or substitution of a single word. The cost for insertions and deletions was set to 1, and 2 for substitutions. Therefore, this metric results in an integer bounded by the length of the longest description in the pair being compared.

- **BiLingual Evaluation Understudy (BLEU) (Papineni et al., 2002).** For each candidate sentence, a modified n-gram precision is calculated with respect to reference lexicalizations. The n-gram lengths usually range from 1 to 4. The metric is adjusted in order to penalize over-generation of common n-grams and favour short and simple sentences. BLEU ranges between 0 and 1.

- **Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (Lin & Och, 2004).** They are a set of measures to determine the quality of an automatically extracted summary by comparing it to other ideal texts created by humans. The measures count the number of overlapping units such as n-gram, word sequences, and word pairs between the computer-generated text to be evaluated and the ideal ones created by humans. We have used the following ROUGE measures:

- *ROUGE-N*, that counts the number of n-grams that match in the texts being compared. The n-gram lengths usually range from 1 to 4.
- *ROUGE-L*, based on longest common subsequence (LCS) statistics between a candidate and a reference text.
- *ROUGE-SN*, that takes into account bigrams that do not have to be consecutive in the text, but could present a maximum of N terms between them.

As an example, Table 2 presents the evaluation results obtained for two pairs of compared sentences:

(a)	A big green desk	(b)	A big green desk
	A big green desk		A big red desk

When both sentences are equal (a) the edit distance is 0 (no edit operations required) and all the BLEU and ROUGE measures are 1 as all n-grams are equal. However, when one word is different (b) the edit distance is 2 (one deletion and one addition are required). This change is also reflected in the n-gram metrics. For example, there are no correct 3- or 4-grams, so BLEU-{3,4} and ROUGE-{3,4} results are 0.

These metrics must be used with care. It has been observed that there is a lack of correlation between automatic metrics and extrinsic measurements more oriented towards the task being accomplished (Gatt & Belz, 2010; Reiter & Belz, 2009). However, we consider that the chosen metrics (string-edit distance, BLEU and ROUGE) are appropriate for this work as we are concerned with whether taking individual author preferences gives rise to a better match (on average) to a corpus that represents outputs by those authors. This task can be accomplished by comparison with a multi-authored corpus as TUNA using automatic metrics that compute how similar two texts are in syntactic and lexical terms. The quality of the generated references in terms of content or task fulfillment are beyond the scope of this work.

Although both BLEU and ROUGE are intended to be calculated against multiple reference outputs they have also been used when only one reference output is available, not only in the TUNA Shared Tasks (Belz & Gatt, 2007; Gatt et al., 2008a, 2009) but also in other works like (Reiter & Belz, 2009) or (Gatt & Belz, 2010).

	String-edit distance	BLEU-				ROUGE-					
		1	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**L**	**S4**
a	0	1	1	1	1	1	1	1	1	1	1
b	2	0.75	0.50	0	0	0.75	0.33	0	0	0.75	0.5

Table 2: Examples for the evaluation metrics

3. Case-Based Reasoning

Case-Based Reasoning (CBR) (Agnar & Enric, 1994) is a paradigm for problem resolution that relies on specific knowledge derived from past experiences. Each problem is considered as a case from the domain, and a new problem is solved by *retrieving* one or more past cases from a case base, *reusing* them somehow, *revising* the solutions obtained, and *retaining* the new experience by including it in the case base.

Reasoning by reusing past experiences is frequently applied to solve problems. Several studies (Schank, 1982; Anderson, 1983) have given empirical evidence for the dominant role of specific, previously experienced situations in human problem-solving.

In CBR terminology, a *case* usually denotes a problem situation. A previously experienced situation, which has been captured and learned in a way that can be reused in the solving of future problems, is referred to as a *past case* or *retained case*. Correspondingly, a *new case* or *unsolved case* is the description of a new problem to be solved. Case-based reasoning is therefore a cyclical and integrated process of solving a problem, learning from this experience, solving a new problem, and so on.

3.1. CBR Cycle

The general CBR cycle may be described by the following four processes:

1. *RETRIEVE* the most similar case or cases. This task starts with a (partial) problem description, and ends when the best-matching previous case has been found.
2. *REUSE* the information and knowledge in the case retrieved to solve the problem. The reuse of the retrieved case solution in the context of the new case focuses on two aspects: the differences between the past and the current case and what part of a retrieved case can be transferred to the new case.

3. *REVISE* the proposed solution. This phase evaluates the case solution generated by reuse. If successful, it learns from this success. Otherwise, it repairs the case solution using domain-specific knowledge.
4. *RETAIN* the parts of this experience likely to be useful for future problem-solving. This is the process of incorporating what is useful to retain from the new problem-solving episode into the existing knowledge.

3.2. Case Retrieval Nets: a Model for Storing the Case Base

Case Retrieval Nets (CRNs) (Lenz & Burkhard, 1996) are a memory model developed to improve the efficiency of the retrieval task in the CBR cycle. As its name indicates, CRNs are organized in nets. The most fundamental item in the context of the CRNs is Information Entities (IEs). These represent any basic knowledge item in the form of an attribute-value pair. A case then consists of a set of such IEs, and the case base is a net with nodes for the IEs observed in the domain and additional nodes denoting the particular cases. IE nodes are connected by similarity arcs, and a case node is reachable from its constituting IE nodes via relevance arcs. Different degrees of similarity and relevance are expressed by varying arcs weights. Given this structure, case retrieval is carried out by activating the IEs given in the query case, propagating this activation according to similarity through the net of IE nodes, and collecting the activation achieved in the associated case nodes.

An example of a Case Retrieval Net applied to the domain of travel agencies is shown in Figure 1. Rectangles represent entity nodes, with their corresponding attribute-value pairs. Hexagons are case nodes, with the description that identifies them univocally. Entity nodes are related among themselves by arcs with black arrowheads, and they are related with cases by arcs with white arrowheads. Weights associated to arcs are not represented in the figure, and arcs with zero weight are omitted.

CRNs present two important features that make them especially suitable for the task we are undertaking. First, CRNs can handle partially specified queries without loss of efficiency, in contrast to most case retrieval techniques which have problems with partial descriptions. These partial queries are very frequent when dealing with lexicalization of information from a conceptual representation. Not always will all sentences have the same structure (e.g. some may have complements and others may not), and in the specific case of referring expressions, not all of them may contain the same attributes when being lexicalized. Second, in CRNs cases do not need to be described

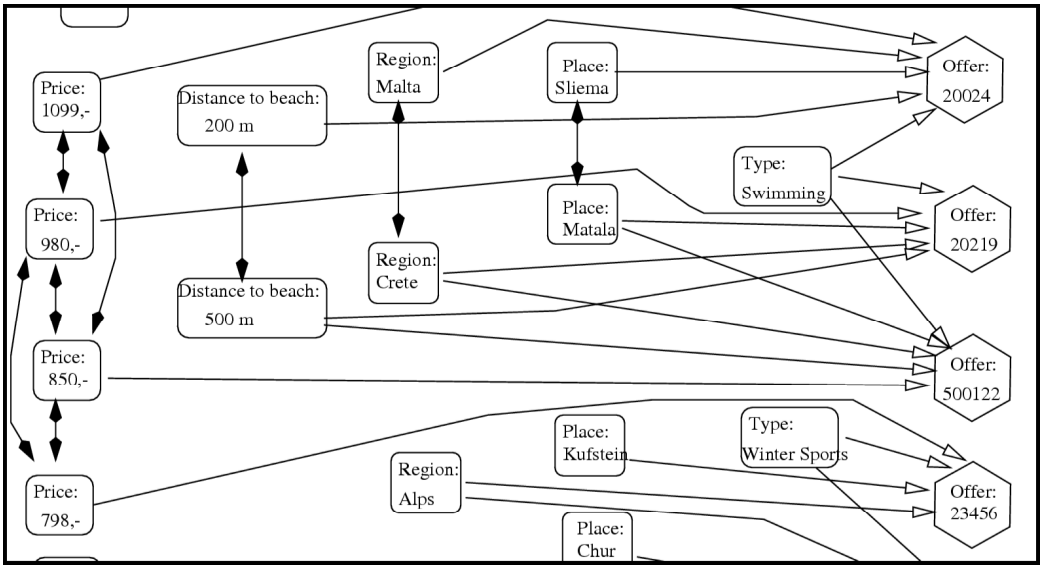

Figure 1: An Example Case Retrieval Net in the domain of travel agencies

by attribute vectors. There are features that would be relevant for some cases but not for others. Many CBR solutions describe exactly what the elements we can expect in any case are. But in the case of language, and more specifically referring expressions, each case can have different elements or types of information to be conveyed in the final text.

4. A CBR Approach to Referring Expression Lexicalization

In order to reproduce different sets of personal preferences for referring expression lexicalization it seems quite natural to use a Case-Based Reasoning approach where past referring expressions can be reused for lexicalizing new ones. Note that when dealing with referring expression lexicalization we are in fact performing two different tasks: syntactic choice (choice of a syntactic form for the referring expression) and lexical choice (choice of a specific word for a piece of information). We are not addressing the selection of attributes in this work, and the given attributes for a referring expression are the input to our lexicalization module. As case base, we have used the two domains of the TUNA corpus: people and furniture.

4.1. The Case Base

As a model for storing the case base, we have used a Case Retrieval Net. This model is appropriate because our cases are formed by attribute-value pairs (elements `ATTRIBUTE` inside the `ATTRIBUTE-SET` from the TUNA corpus), and also because the queries to the module will not always have the same elements. When the system is asked for the lexical realization of a new referring expression, it looks for other referring expressions related to the set of attributes that define them.

In our approach, a case consists of a description of the problem (`ATTRIBUTE-SET`) and a solution to this problem (`DESCRIPTION` considered as a text template). For each attribute-value pair in the `ATTRIBUTE-SET` an Information Entity (IE) is created, and for each case a node containing links to the IEs that compose the case is created too. Each `ATTRIBUTE` element in `DESCRIPTION` is considered as a slot of the template that will be later completed with the corresponding information. `DESCRIPTIONs` are therefore considered as lexicalization solutions to the cases. An example can be seen in Figure 2 and Table 3.

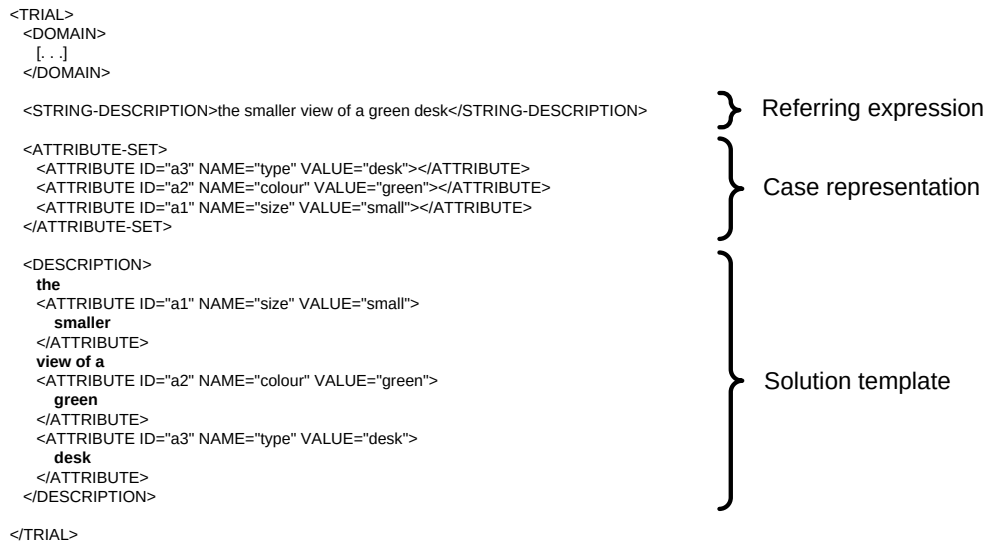

Figure 2: Example of case from the corpus

As the different entities are inserted while composing the net, similarities have to be established between them. Taking into account that the available values for each of the attributes are not usually related (the color does not

have anything in common with wearing a tie, for example), different similarities have been established depending on the specific type of attribute. In addition, the similarity between equal values of the same attribute will always be 1 (the maximum possible) and the similarity between values of different attributes will always be 0. The rest of defined similarities are the following:

- `type` and `colour`. All the values that are not equal have a similarity of 0.2. It was decided to use a non-zero similarity because at least all the values are of the same type (different kinds of furniture or different colours), and this is always more similar than being of different types.
- `orientation`. Similarities were established by considering the relative distance between the four values (*left*, *right*, *front* and *back*). Adjacent values have a similarity of 0.5 (as for example *left* and *front*), and non-adjacent values a similarity of 0 (as in the case of *left* and *right*).
- `x-dimension` and `y-dimension`. The similarity between the two values is calculated as the relative distance between them, considering the total number of rows and columns using Formulas 1 and 2. The similarity between equal values will always be 1, but for example in the case of x=2 and x=4 it will be 0.4, or with y=1 and y=2 it will be 0.33.

$$sim_x(x_1, x_2) = 1 - \frac{|x_1 - x_2|}{5} \tag{1}$$

$$sim_y(y_1, y_2) = 1 - \frac{|y_1 - y_2|}{3} \tag{2}$$

- The rest of the attributes (`size`, `age`, `hairColour`, `hasShirt`, `hasBeard`, `hasHair`, `hasGlasses`, `hasTie` and `hasSuit`) have values that are always opposite (*young* vs. *old*, 0 vs. 1, etc.), so the similarity between different values of these attributes will always be 0.

Template	**Attribute-Value**	**Lexicalization**
The ____ (size) view of a _____ (colour) ____ (type)	size-small	*smaller*
	colour-green	*green*
	type-desk	*desk*

Table 3: Template and preferred lexicalizations obtained from case in Figure 2

Similarities were determined in an intuitive way, but did not consider that people may find colours like *blue* and *green* more similar than *blue* and *red*, or opposite orientations more similar than adjacent ones. More work about how people perceive the similarities between these values will be addressed in future work.

4.2. CBR Cycle for Lexicalization of Referring Expressions

Each process in the CBR cycle is explained in the following subsections.

4.2.1. Case Retrieval

Attribute-value pairs from the `ATTRIBUTE-SET` that has to be lexicalized are considered as the query. In our module the retrieval of cases is directly handled by the Case Retrieval Net and its method of similarity propagation. Starting from the attributes and values that we need to lexicalize, the retrieval of the most similar cases is done by calculating an activation value for each case in the case base. The cases retrieved with higher activation are more similar to the given query and are ordered by preference taking into account the attributes they contain. The system organizes these cases into four different groups from higher to lower preference:

1. *Matching case*. Cases that contain exactly the same attributes as the query.
2. *Encompassing case*. Cases that contain all the attributes in the query, and some additional ones.
3. *Included case*. Cases that lack some attributes from the query and have no extra ones.
4. *Overlapping case*. Cases that lack some attributes from the query, but have some extra ones that were not in the query.

A graphical representation of each situation is presented in Figure 3.

The cases with maximum activation are classified using these groups, and the order given is the preferred order to choose the most suitable case for the query. If more than one case from the same group have the maximum activation, the retrieved case is chosen randomly among them (more refined strategies like favoring the case that is easiest to adapt or that has minimal information loss will be addressed in the future).

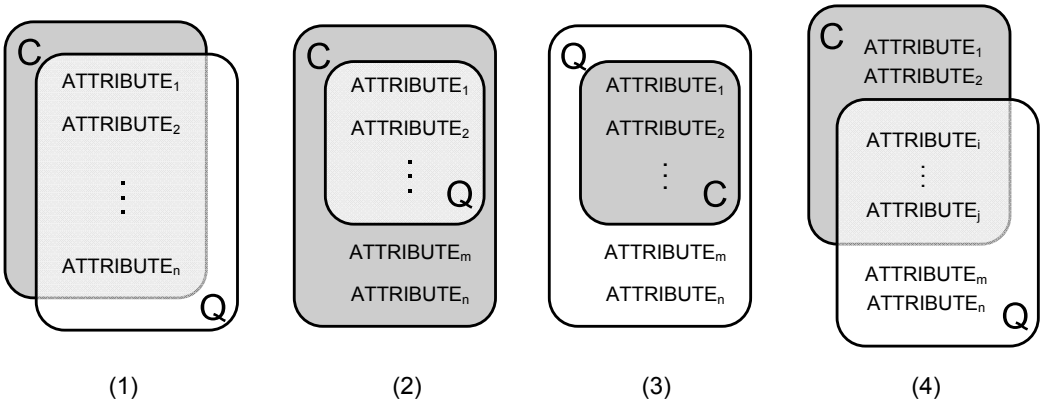

Figure 3: Classification of a case C depending on its relation to query Q: matching case (1), encompassing case (2), included case (3) and overlapping case (4)

4.2.2. Case Adaptation or Reuse

The adaptation of the chosen case depends on its type. The idea is to retain all the parts of the template that correspond to attributes common to the query and the case. Extra attributes in the case that do not appear in the query are discarded. Attributes in the query not appearing in the case are lost.

In order to adapt each of the attributes inside the template, the system checks whether the values of each attribute are the same for the query and the case retrieved. If they are, the lexical expression used in the case retrieved is used for the new referring expression. If not, a list of default expressions is available for each possible value of an attribute. These default expressions are the ones most frequently used in the corpus. Determiners or other parts of the template that do not correspond to any of the attributes are always included.

4.2.3. Case Revision and Retainment

Once a case solution has been generated, whether correctly or not, an opportunity to learn arises. At the moment, this task is not implemented in our CBR module. It would be very useful to incorporate new knowledge in the case base, but when dealing with natural language this could be a very challenging task. Due to the constraints associated with language use, the presence of an expert in the domain is required to revise the achieved results achieved from the module, retaining and refining them if possible. In

addition, if the idea is to generate text following a specific style, this task is even more complex.

4.3. An Example of System Operation

An example is given to show how the algorithm works. Suppose that the algorithm is presented with a query like:

```
TYPE:   COLOUR: Y-DIMENSION:
chair   grey    2
```

The net retrieves a case from the ones considered *matching cases* because it contains exactly the same attributes as the query:

```
TYPE:   COLOUR: Y-DIMENSION:
chair   grey    3
```

with *the grey chair in the bottom row* as associated referring expression, and corresponding to the following template:

"the"
<ATTRIBUTE name=colour value=grey string=***"grey"***/>
<ATTRIBUTE name=type value=chair string=***"chair"***/>
<ATTRIBUTE name=y-dimension value=3 string=***"in the bottom row"***>

In order to adapt the template retrieved we must check whether the values in the case retrieved match the values in the query. If so, we maintain the lexical tag associated with that value. Otherwise we use a default tag that is assigned to each value. In this example, the values *grey* and *chair* match, but not the *y-dimension* one. Instead of using *in the bottom row* we use the default text for value 2: *in the middle row*. The final referring expression obtained would be: *the grey chair in the middle row*.

5. Evaluation of Personal Preferences

We tested the system explained in Section 4 following two different approaches: one that takes personal preferences into account, and another that

does not. As a first step towards the comparison of the different results obtained by the system, we studied the lexical variation demonstrated by the corpus. If there were no lexical variation for the attributes being lexicalized, personal preference would not influence the results obtained. We also analyzed the performance of the algorithm in terms of the type of case retrieved and the usage of default values in the lexicalizations.

5.1. Lexical Variation in the TUNA Corpus

The TUNA corpus contains 398 examples for the furniture domain and 340 for the people domain, with 6 and 12 attributes per element respectively. Table 4 shows the lexical variation found for the different attributes in both domains (number of different lexicalizations used per attribute vs. total number of mentions to this attribute). The number of possible values for each attribute is also included in the table. Note that due to the nature of the corpus we are considering as lexicalizations both lexical and syntactic structures used for the attributes. For example, the value "big" was sometimes lexicalized using single words or expressions (e.g. *large*), and sometimes using phrases (e.g. *that is big*). Therefore, we are considering referring expression lexicalization as both lexical choice and syntactic choice as they are interlaced in the TUNA corpus.

For each attribute the distribution of lexicalizations is homogeneous for the different values. The only exception is the value *grey* of the attribute *colour* in the furniture domain. Whereas the other colours have an average of 2 lexicalizations in 70 mentions, the grey colour presents 10 different lexicalizations in 91 mentions. Related to this lexical variation we find the problem of ambiguity in annotation. Usage of grey in TUNA corpus annotations is overloaded, being used indistintly to indicate various meanings such as *black and white*, *monochrome* or *silver*. More than alternative lexicalizations that could be used indistinctly, these are cases of loss of information during the annotation process, as *silver* or *black and white* could be used sometimes (but not always) as a synonym of grey.

It is interesting to note that the attributes *orientation*, *x-dimension* and *y-dimension* present high variation in both domains. In the people domain attributes *hasShirt*, *hasSuit* and *hasTie* have a 100% variation, with a different lexicalization for each of their mentions in the corpus (although the number of mentions is low). All those attributes with high lexical variation correspond to features that do not have an easy and fixed lexicalization as could be the case of colours.

Attribute	Different values	Different lexicalizations	Total mentions	% of variation
Furniture				
orientation	4	79	127	62 %
x-dim	5	59	105	56 %
y-dim	3	53	132	40 %
size	2	17	130	17 %
type	4	22	371	6 %
colour	4	17	344	5 %
People				
hasShirt	2	3	3	100 %
orientation	3	9	9	100 %
hasSuit	2	4	4	100 %
hasTie	2	3	3	100 %
hasHair	2	5	9	56 %
x-dim	5	63	106	59 %
y-dim	3	55	122	45 %
age	2	7	21	33 %
hasBeard	2	16	76	21 %
hasGlasses	2	19	140	14 %
hairColour	3	11	104	11 %
type	1	11	284	4 %

Table 4: Lexical variation in the TUNA corpus

When analyzing the corpus separately by its individual authors, variation is quite low. It seems that authors are fairly consistent with themselves, tending to use the same kind of lexicalizations. Table 5 presents the average variation by author for both furniture and people. The total number of mentions for each attribute is added for easier comparison of results. An average variation of 0% means that none of the authors was *inconsistent* with themselves, that is, everytime they used the same value for an attribute they used the same lexical expression. This kind of coherence (as inverse to variation) is only perfect in attributes like *hasHair* or *hasTie* that were barely used. All authors present a degree of variation lower than 50%, and most of them are between 0% and 25%. This data is presented in Table 6. It is also interesting to note that authors are more consistent in the furniture domain, probably due to furniture items being obviously artificial

(they were digital representations of objects), whereas people items were real photographs. This could have helped to make authors rely on stereotypical verbalisations of "basic" visual properties like size and colour.

Furniture			**People**		
Attribute	**Avg. var. by author**	**Total mentions**	**Attribute**	**Avg. var. by author**	**Total mentions**
y-dim	25%	132	*hasGlasses*	37%	140
x-dim	10%	105	*hasBeard*	26%	76
size	8%	130	*hasShirt*	25%	3
orientation	5%	127	*y-dim*	23%	122
colour	4%	344	*x-dim*	15%	106
type	3%	371	*age*	13%	21
			orientation	13%	9
			type	12%	284
			hairColour	4%	104
			hasHair	0%	9
			hasSuit	0%	4
			hasTie	0%	3

Table 5: Lexical variation in the TUNA corpus considering individual authors

Furniture		**People**	
% of variation	**Num. of authors**	**% of variation**	**Num. of authors**
Exactly 0%	18	Exactly 0%	8
(0%, 10%]	22	(0%, 10%]	9
(10%, 25%]	15	(10%, 25%]	28
(25%, 50%]	2	(25%, 50%]	12
More than 50%	0	More than 50%	0
Total:	57	**Total:**	57

Table 6: Distribution of authors depending on their variation

The high lexical variation shown by the TUNA corpus, which only contains two different domains and a limited number of attributes for each of them, reinforces the idea of how complex it is to select the appropriate lexical items when lexicalizing language in general and referring expressions in particular. However, it seems that this variation is due to differences between different authors as they tend to be consistent with themselves.

5.2. Experiments

For the approach without taking into account any kind of personal preferences, we used all the examples in the corpus at the same time (separated by domains). In our second approach we tried to reproduce the lexicalization preferences of different people so we divided the corpus by its individual authors. Here we faced the problem of each author having only a few examples in the corpus (seven for furniture and six for people).

For the approach without taking into account personal preferences we divided the complete corpus in random sets of seven and six examples (for furniture and people respectively) in order to have results easily comparable with the second approach. We then performed a 7-fold cross validation for the furniture domain and a 6-fold cross validation for the people domain for each of these sets, so each example in the corpus was generated by the CBR algorithm trained with only six (furniture) or five (people) examples. The results obtained were measured using string-edit distance, BLEU and ROUGE (see Section 2.3.2). They are shown in Table 7.

An important issue is how to resort to default lexicalizations when the values of the case retrieved and the query are not the same. When no personal data is taken into account, the default values for each attribute and value are computed over the complete corpus. When the system is working over each author's data, the default values used correspond to those belonging to this specific author.

The results obtained for this initial system are rather poor when it comes to matching the specific expressions used in the corpus. We consider this to be due to the fact that the nature of the corpus indicates a broad variation in the type of expression used, aimed at identifying a number of possible ways of describing referents as actually done by human authors, rather than setting the correct way of referring. In this way, the same set of attributes can be lexicalized in a varied set of expressions, and most times the chosen lexicalization is correct but does not correspond to the original one in the corpus. Another possible reason could be that when considering the whole corpus, there are several *ties* that are returned as the best matches, and in these cases the best case is chosen randomly.

In our second approach we tried to reproduce the lexicalization preferences of the different authors of the corpus. We divided the corpus by authors, and for the set of examples created by each of them we also performed a cross validation. A 7-fold cross validation was performed for the furniture domain and a 6-fold cross validation for the people domain. A Wilcoxon

	Furniture	People
String-edit distance	5.11	6.56
BLEU-1	0.43	0.41
BLEU-2	0.30	0.26
BLEU-4	0.13	0.10
ROUGE-1	0.63	0.56
ROUGE-2	0.37	0.20
ROUGE-4	0.02	0.00
ROUGE-L	0.61	0.52
ROUGE-S4	0.33	0.25

Table 7: Evaluation results with the system working over the complete data

Signed Ranks Test was performed in order to test if differences between both versions of the system were reliably different. Results are shown in Table 8.

	Results		Significance (Wilcoxon Test)	
	Furniture	People	Furniture	People
String-edit dist.	2.05	3.71	(Z = -14.56, p $<$ 0.001)	(Z = -11.69, p $<$ 0.001)
BLEU-1	0.80	0.69	(Z = -14.97, p $<$ 0.001)	(Z = -11.71, p $<$ 0.001)
BLEU-2	0.73	0.57	(Z = -14.59, p $<$ 0.001)	(Z = -11.03, p $<$ 0.001)
BLEU-4	0.61	0.42	(Z = -10.88, p $<$ 0.001)	(Z = -8.25, p $<$ 0.001)
ROUGE-1	0.87	0.75	(Z = -12.82, p $<$ 0.001)	(Z = -9.37, p $<$ 0.001)
ROUGE-2	0.69	0.43	(Z = -12.19, p $<$ 0.001)	(Z = -8.14, p $<$ 0.001)
ROUGE-4	0.19	0.04	(Z = -7.79, p $<$ 0.001)	(Z = -3.94, p $<$ 0.001)
ROUGE-L	0.84	0.71	(Z = -12.84, p $<$ 0.001)	(Z = -9.57, p $<$ 0.001)
ROUGE-S4	0.68	0.48	(Z = -12.48, p $<$ 0.001)	(Z = -8.63, p $<$ 0.001)

Table 8: Evaluation results with the system working over each author's data

As we can observe from the results, the system that takes into account the preferences of the authors reduced the string-edit distance in 59.88% for the furniture domain and 43.45% for the people domain. Those values are quite high, reflecting the importance of taking personal preferences into account when dealing with the lexicalization of referring expressions. Improvement in the BLEU and ROUGE values also supports these observations: there is a higher coincidence of n-grams when personal preferences are considered. The Wilcoxon Test results show that all differences between both approaches are statistically significant (p $<$ 0.001).

The distribution of the type of cases retrieved by the system can be seen in Table 9. Both approaches present similar numbers. For the furniture domain

most of the cases retrieved are of the *matching* type (cases containing exactly the same attributes as the query) and the *encompassing* type (cases with the same attributes as the query and some more). And for the people domain in the approach that does not take personal preferences into account there are more *encompassing* and *overlapping* cases whereas in the second approach the number of *matching* cases is higher. These results are expected if we consider the larger number of possible attributes in the people domain. It is likely that different people will use different subsets of attributes (increasing the number of *overlapping* and *encompassing* cases when preferences are not taken into account), though any given person will manifest personal preferences in the choice of attributes (hence we have more *matching* cases when such preferences are taken into account). Considering the performance of the CBR system, the high number of *overlapping* and *included* cases is expected given the low number of examples used for training. From the point of view of preference considerations, *matching* and *encompassing* cases are more used when personal preferences are taken into account. This is good as these two types are the only ones where there is no loss of information from the query, and it is due to the more similar cases in the training set when it is generated with the examples made by an individual author.

No personal preferences		
Case Type	**Furniture**	**People**
Matching case	126 (31.66%)	89 (26.18%)
Encompassing case	126 (31.66%)	101 (29.70%)
Overlapping case	82 (20.60%)	109 (32.06%)
Included case	64 (16.08%)	41 (12.06%)
Personal preferences		
Case Type	**Furniture**	**People**
Matching case	137 (34.42%)	141 (41.47%)
Encompassing case	159 (39.95%)	89 (26.18%)
Overlapping case	37 (9.30%)	82 (24.12%)
Included case	65 (16.33%)	28 (8.23%)
Total	398	340

Table 9: Distribution of the type of cases retrieved. Percentages over the total are shown in parentheses

Another important issue is how often the system has to resort to default lexicalizations when the values of the case retrieved and the query are not

the same. The percentages of default lexicalizations used for both approaches can be seen in Table 10. Again results for both approaches are quite similar. Default lexicalizations must be used frequently especially in the furniture domain. Again, it is interesting to note that even when the lexical tags for certain values cannot be reused, the influence of the syntactic structures and the default values in the templates produces much better results in the second approach.

No personal preferences		
	Furniture	**People**
Number of lexicalizations	1312	1084
Default lexicalizations	455	147
Default percentage	34.68%	13.56%
Personal preferences		
	Furniture	**People**
Number of lexicalizations	1338	1057
Default lexicalizations	651	139
Default percentage	48.65%	13.15%

Table 10: Percentage of default lexicalizations

5.3. Some Examples

Here we show a few examples of the referring expressions generated by the system with and without personal preferences. We will start with the following query:

```
TYPE:   ORIENTATION:
chair   right
```

that corresponds to the referring expression *a chair facing to the right* in the corpus.

If we do not consider personal preferences, the net retrieves a case from the *encompassing case* group (with more attributes than the query):

```
TYPE:   COLOUR: ORIENTATION:
chair   red     right
```

with *red dining chair at oblique angle facing right* as the associated referring expression. With this retrieved case the referring expression produced is

dining chair at oblique angle facing right, which is not very similar to the one corresponding to the original query. Take note that the use of determiners is different, the chair is considered as a *dining chair*, and it is at an *oblique angle facing right* when it was facing right.

Taking into account personal preferences, the net retrieves one of the cases corresponding to the same user that is also of type *encompassing case*:

```
TYPE:   COLOUR: ORIENTATION:
chair   grey    back
```

with *a grey chair facing backwards* as the associated referring expression. In this case there are two values that differ from the query (*grey* and *back*). However, the referring expression produced (*a chair facing right*) is much more similar to the original in the use of determiners and the orientation expression.

In order to exemplify the algorithm in the people domain, the algorithm receives a query like:

```
TYPE:   Y-DIMENSION:
person  1
```

that corresponds to the referring expression *top man* in the corpus.

Not considering personal preferences, the net retrieves an *encompassing* case, with more attributes than the query:

```
TYPE:   HAS_BEARD: Y-DIMENSION:
person  true       1
```

with *male subject on top with beards* as the associated referring expression. The referring expression obtained after adaptation (*male subject on top*) is very different from the original one.

If we take into account personal preferences, the net also retrieves an *encompassing* case from the ones that correspond to the same user:

```
TYPE:   X-DIMENSION:   Y-DIMENSION:
person  5              1
```

with *last top man* as the associated referring expression. After adaptation the referring expression *top man* is generated. In this case both the original

and the obtained expressions are the same even when the cases were quite different. This is because both cases show the same preferences in syntax and vocabulary (short sentence without prepositions and the use of the word *man* for the type attribute).

5.4. Discussion

Some strategies developed for referring expression generation are related to this work as they have considered the personal preferences of different people (see Section 2.2). Although those systems used similar approaches to the generation and lexicalization of referring expressions, none of them have compared the impact of personal preferences on these tasks (Bohnet, 2008, 2009; Di Fabbrizio et al., 2008). However, it would be interesting to know whether their systems would have obtained the same results even if they had not taken into account the personal preferences of the authors. Their results are presented in Table 11 for comparison with our own results. Note that data is not directly comparable as their results are computed over a 20% of the TUNA corpus (the remaining 80% was used for training). We have not added to the comparison the results over the final test data as it was newly created for the tasks and it does not belong to the TUNA corpus. In addition, we can only compare the results of the string-edit distance metric as it is the only one the participants presented in their reports. In the cases where the participants presented more than one system, we have taken only the one with better results. As we can see in the table, our approach outperforms the other systems in both domains, with the only exception of (Bohnet, 2008) in the people domain.

	Furniture	**People**
(Bohnet, 2008)	3.16	3.65
(Di Fabbrizio et al., 2008)	3.52	4.25
(Bohnet, 2009)	3.86	4.76
Our approach	2.05	3.71

Table 11: String-edit distance results for other participants in the shared tasks in comparison with our own results

It is not the first time that CBR techniques have been used for referring expression generation. Both Kelleher & Mac Namee (2008) and Hervás & Gervás (2009) applied case-based reasoning for the lexicalization of referring

expressions. Using each of the situations from the TUNA corpus as a template, Kelleher and Mac Namee relied on similarity between cases in both the attributes and their values in the templates, whereas Hervás and Gervás considered the presence of attributes important and took into account the values in a more refined adaptation of the cases to create the solution.

6. Conclusions and Future Work

The data presented in this paper displays the high lexical variability of natural language in general and the TUNA corpus in particular. We have demonstrated that the performance of a lexicalization algorithm that tries to imitate human-generated referring expressions improves greatly by modeling particular preferences. Our case-based approach was easily adapted to model this personalization effort by narrowing the training set to only those descriptions produced by a single person and having the software model that particular way of generating referring expressions.

For this solution the CBR approach turned out to be quite appropriate, as the case base could store only the descriptions created by a particular person, and during lexicalization different referring expressions could be generated but always maintaining the same style. However, sometimes the system is not capable of finding a perfect match for a given query, but a solution with more attributes than the query, or one that lacks some of its attributes. This could be avoided by improving the reuse stage of the CBR module. In the former case the resulting adaptation is a partial solution to the problem posed by the query. A secondary retrieval process could be set in motion, using as a query simply the set of attributes in the query that could not be accommodated in the partial solution provided by the first case retrieved by the system. In the latter case, there will be vacant attributes in the corresponding solution. The easiest solution is to keep the values of the past case in the slots for which the query does not specify any value. Better results can be obtained by consulting the system knowledge base for concepts that the knowledge base shows as related to those appearing in the query. In order to be appropriate as fillers for the vacant slots, these concepts must be within a given threshold of similarity with respect to the original values given in the retrieved case for those attributes.

Our approach, and other similar ones already mentioned, are based on the idea of each person having specific preferences. However, all of them miss the broader idea of preference features that can be shared by different people.

For example, many people will share preferences in the use of determiners or the choice of vocabulary for certain concepts. In future work we will study the distribution of preferences across multiple individuals. Those preferences or styles could then be used to lexicalize referring expressions by using the shared preferences corresponding to a set of different people, instead of using only the preferences of a single person. This could be important in applications like dialog systems, where the system must try to accommodate its vocabulary and means of expression to those shown by the user (Pickering & Garrod, 2004). Style-based approach to the lexicalization of referring expressions could then be used to detect the style of expression the user is employing, and then adapt system responses to that style. The style detection would be especially important in adapting systems of this kind as the system is unlikely to have information about who it is communicating with. Recently, style has started to be researched more intensively, not only at the level of Referring Expression Generation, but also globally (Paiva & Evans, 2005; Mairesse & Walker, 2011).

Acknowledgments

This research is funded by the Spanish Ministry of Education and Science (TIN2009-14659-C03-01) and Universidad Complutense de Madrid (GR58/08). We thank the editors and reviewers, as well as Mark Finlayson and Pablo Moreno, for their helpful comments and discussion.

References

Agnar, A., & Enric, P. (1994). Case-based Reasoning : Foundational issues, methodological variations, and system approaches. *AI Communications*, *7*.

Anderson, J. R. (1983). *The architecture of cognition*. Harvard University Press.

Bangalore, S., & Rambow, O. (2000). Corpus-based lexical choice in natural language generation. In *38th Meeting of the Association for Computational Linguistics (ACL'00)*.

Belz, A., & Gatt, A. (2007). The Attribute Selection for GRE Challenge: Overview and evaluation results. In *Proc. 2nd UCNLG Workshop: Language Generation and Machine Translation (UCNLG+MT)* (pp. 75–83). Copenhaguen, Denmark.

Bohnet, B. (2008). The Fingerprint of Human Referring Expressions and their Surface Realization with Graph Transducers. In *Referring Expression Generation Challenge 2008, Proc. of the 5th International Natural Language Generation Conference (INLG'08)*.

Bohnet, B. (2009). Generation of Referring Expression with an Individual Imprint. In *Generation Challenges 2009, Proc. of the European Natural Language Generation Conference 2009 (ENLG'09)*.

Cahill, L. (1998). *Lexicalisation in applied NLG systems*. Technical Report ITRI-99-04.

Clements, M., de Vries, A., & Reinders, M. (2010). The influence of personalization on tag query length in social media search. *Information Processing & Management*, *46*, 403 – 412. Semantic Annotations in Information Retrieval.

Dale, R., & Haddock, N. (1991). Generating referring expressions involving relations. In *Proceedings of the fifth conference on European chapter of the Association for Computational Linguistics* (pp. 161–166). Morristown, NJ, USA: Association for Computational Linguistics.

Dale, R., & Reiter, E. (1995). Computational Interpretation of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, *19*, 233–263.

van Deemter, K., van der Sluis, I., & Gatt, A. (2006). Building a semantically transparent corpus for the generation of referring expressions. In *Proceedings of the 4th International Conference on Natural Language Generation (Special Session on Data Sharing and Evaluation), INLG-06*.

Di Fabbrizio, G., Stent, A., & Bangalore, S. (2008). Referring Expression Generation Using Speaker-based Attribute Selection and Trainable Realization (ATTR). In *Referring Expression Generation Challenge 2008, Proc. of the 5th International Natural Language Generation Conference (INLG'08)*.

Edmonds, P., & Hirst, G. (2002). Near-Synonymy and Lexical Choice. *Computational Linguistics*, (pp. 105–144).

Elhadad, M. (1992). *Using argumentation to control lexical choice: a functional unification-based approach*. Phd dissertation Department of Computer Science, Columbia University.

Elhadad, M. (1993). *FUF: The universal unifier. User manual, version 5.2.*. Technical Report CUCS-038-91 Columbia University.

Gatt, A. (2007). *Generating Coherent References to Multiple Entities*. Ph.D. thesis.

Gatt, A., & Belz, A. (2010). Introducing shared task evaluation to nlg: The tuna shared task evaluation challenges. In E. Krahmer, & M. Theune (Eds.), *Empirical Methods in Natural Language Generation*. Springer volume 5790 of *Lecture Notes in Computer Science*.

Gatt, A., Belz, A., & Kow, E. (2008a). The TUNA Challenge 2008: Overview and evaluation results. In *Proceedings of the 5th International Conference on Natural Language Generation, INLG-08* (pp. 198–206). Ohio, USA.

Gatt, A., Belz, A., & Kow, E. (2009). The TUNA-REG Challenge 2009: overview and evaluation results. In *ENLG '09: Proceedings of the 12th European Workshop on Natural Language Generation* (pp. 174–182). Morristown, NJ, USA: Association for Computational Linguistics.

Gatt, A., van der Sluis, I., & van Deemter, K. (2007). Evaluating algorithms for the generation of referring expressions using a balanced corpus. In *Proc. of the 11th European Workshop on Natural Language Generation (ENLG 07)*.

Gatt, A., van der Sluis, I., & van Deemter, K. (2008b). *XML Format Guidelines for the TUNA Corpus*. Technical Report University of Aberdeen.

Hervás, R., & Gervás, P. (2008). Degree of Abstraction in Referring Expression Generation and its Relation with the Construction of the Contrast Set. In *Proc. of the 5th International Natural Language Generation Conference (INLG'08)*.

Hervás, R., & Gervás, P. (2009). Evolutionary and Case-Based Approaches to REG: NIL-UCM-EvoTAP, NIL-UCM-ValuesCBR and NIL-UCM-EvoCBR. In *Generation Challenges 2009, Proc. of the European Natural Language Generation Conference 2009 (ENLG'09)*.

Horacek, H. (1997). An algorithm for generating referential descriptions with flexible interfaces. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*.

Janarthanam, S., & Lemon, O. (2009). Learning lexical alignment policies for generating referring expressions in spoken dialogue systems. In *ENLG '09: Proceedings of the 12th European Workshop on Natural Language Generation* (pp. 74–81). Morristown, NJ, USA: Association for Computational Linguistics.

Kelleher, J., & Mac Namee, B. (2008). Referring Expression Generation Challenge 2008 DIT System Descriptions. In *Referring Expression Generation Challenge 2008, Proc. of the 5th International Natural Language Generation Conference (INLG'08)*.

Krahmer, E., van Erk, S., & Verleg, A. (2003). Graph-Based Generation of Referring Expressions. *Computational Linguistics*, *29*, 53–72.

Krahmer, E., & Theune, M. (1998). Context sensitive generation of descriptions. In *ICSLP-98* (pp. 1151–1154). Sydney, Australia.

Krahmer, E., & Theune, M. (2002). Efficient context-sensitive generation of referring expressions. In *Information Sharing: Givenness and Newness in Language Processing* (pp. 223–264).

Lenz, M., & Burkhard, H.-D. (1996). Case Retrieval Nets: Basic Ideas and Extensions. In *KI - Kunstliche Intelligenz* (pp. 227–239).

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, *10*, 707–710.

Lin, C.-Y., & Och, F. J. (2004). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics* ACL '04. Stroudsburg, PA, USA: Association for Computational Linguistics.

Lorigo, L., Pan, B., Hembrooke, H., Joachims, T., Granka, L., & Gay, G. (2006). The influence of task and gender on search and evaluation behavior using google. *Information Processing & Management*, *42*, 1123–1131.

Mairesse, F., & Walker, M. A. (2011). Controlling user perceptions of linguistic style: Trainable generation of personality traits. *Comput. Linguist.*, *37*, 455–488.

Paiva, D. S., & Evans, R. (2005). Empirically-based control of natural language generation. In *Proceedings of the ACL 2005 Conference*.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* (pp. 311–318). Morristown, NJ, USA: Association for Computational Linguistics.

Pickering, M., & Garrod, S. (2004). Toward a mechanistic psychology of dialogue. *Behav Brain Sci.*, *27*, 169–90.

Reiter, E., & Belz, A. (2009). An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Comput. Linguist.*, *35*, 529–558.

Reiter, E., & Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press.

Reiter, E., & Sripada, S. (2002). Should corpora texts be gold standards for NLG. In *In Proceedings of the Second International Conference on Natural Language Generation* (pp. 97–104).

Schank, R. (1982). *Dynamic memory; a theory of reminding and learning in computers and people*. Cambridge University Press.

Siddharthan, A., & Copestake, A. (2004). Generating referring expressions in open domains. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics* ACL '04. Stroudsburg, PA, USA: Association for Computational Linguistics.

Stoyanchev, S., & Stent, A. (2009). Lexical and syntactic priming and their impact in deployed spoken dialog systems. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers* NAACL-Short '09 (pp. 189–192). Stroudsburg, PA, USA: Association for Computational Linguistics.

van der Sluis, I., Gatt, A., & van Deemter, K. (2006). *Manual for the TUNA Corpus: Referring expressions in two domains*. Technical Report AUCS/TR0705 University of Aberdeen.