20 Natural Language Generation

EHUD REITER

Natural language generation (NLG) systems generate texts in English and other human languages. Although they are based on many of the same linguistic and algorithmic insights that underlie other kinds of natural language processing systems, they are not simply natural language understanding systems run 'in reverse.'

In this chapter, I will briefly review NLG. In Sections 1–4, I summarize the basic concepts of NLG, and try to highlight issues which are important to building NLG systems and also potentially interesting to linguists and psycholinguists. Then, in Section 5, I discuss some contemporary research in NLG, in areas which are of personal interest to me and which I believe should be of interest to other researchers in the language community. The research topics discussed are by no means exhaustive – they are just a sample of what the NLG research community is working on. I conclude (Section 6) with a brief summary of resources (software, corpora, information) for researchers who wish to work in NLG.

1 High-Level Perspective: Making Choices about Language

From a high-level perspective, perhaps the biggest difference between NLG and other types of NLP is the central role of choice making. NLG systems have to make numerous choices about their output texts, ranging from high-level choices about appropriate content to low-level choices about the use of pronouns. Although choice making of course also occurs in other NLP tasks (for example machine translation systems have to decide which target-language word to use when translating a source-language word), choice making is arguably more central to NLG than to most other areas of NLP.

Sometimes NLG choices can be made on the basis of linguistic correctness, as in the following example of a pronominalization choice:

The Handbook of Computational Linguistics and Natural Language Processing
Edited by Alexander Clark, Chris Fox and Shalom Lappin
© 2010 Blackwell Publishing Ltd. ISBN: 978-1-405-15581-6

(1) a) * *'Mary read about Mary.'*
    b) *'Mary read about herself.'*

In this case the NLG system can use binding theory (Büring 2005) to determine that a reflexive pronoun must be used in this context, and hence decide to generate (1b).

However, in other cases, such as the example below, both choices lead to linguistically correct texts:

(2) a) *'I bought an apple. I ate it.'*
    b) *'I bought an apple. I ate the apple.'*

Since both choices lead to valid texts, the NLG system must decide between them using other criteria. Often such decisions are made based on readability factors, which in turn are based on psycholinguistic models of language comprehension; in the above example such models might suggest that (2a) will be read faster, and hence should be generated. Decisions may also be influenced by genre constraints; for example, pronoun usage may be discouraged in safety-critical texts such as operation manuals for nuclear power plants, so in such genres (2b) should be generated. Genre models are typically based on corpus analysis or explicit genre writing guides. In some cases decisions are also influenced by the linguistic abilities and preferences of the reader of the text; for example an NLG system may choose (2b) if its reader is not fluent in English.

NLG is thus largely the study of choice making, including analyses of individual choices; architectures and systems that can be used to make sets of choices; and methodologies for creating and evaluating new choice making rules and systems. Analyses of individual choices are often based on linguistic and/or psycholinguistic research; analyses of choice making architectures and systems often build on artificial intelligence techniques; and methodologies for creating and evaluating rules draw on both AI and (psycho)linguistics.

2 Two NLG Systems: SumTime and SkillSum

In order to illustrate choice making and other aspects of NLG, it is useful to examine some real NLG systems. In this section we look at SumTime, which generates weather forecasts, and SkillSum, which generates feedback reports on educational assessments.

2.1 SumTime: Weather Forecasts

One popular application of NLG is generating textual weather forecasts from numerical weather prediction data. NLG systems such as FOG (Goldberg et al., 1994), MultiMeteo (Coch 1998), SumTime (Reiter et al., 2005), and RoadSafe (Turner et al., 2008) take as their input a large set of numbers which predict temperature, precipitation, wind speed, and so forth at various points and times.

Table 20.1 Numerical wind forecast for September 19, 2000

Time	*Wind Dir.*	*Wind Speed*
06:00	SE	11
09:00	SSE	13
12:00	SSE	14
15:00	SSE	15
18:00	SE	18
21:00	SE	23
00:00	SE	28

Corpus (human) text:
SSE 10–15 INCREASING 15–20 BY EVENING AND 25–30 LATER.

SumTime text:
SE 9–14 veering SSE 13–18 by mid-afternoon, then increasing SE 26–31 by midnight.

Figure 20.1 Human and corpus wind descriptions for September 19, 2000.

These numbers are produced by a supercomputer running a numerical weather simulation, and modified by human forecasters based on their knowledge of local meteorological conditions. From this input data, the systems produce weather forecast texts (sometimes in multiple languages) which are sent to forecast users (usually after being checked and post-edited by the human forecasters) (Sripada et al., 2004). Table 20.1 shows a simple extract from one of SumTime's data sets, showing 24 hours of predicted wind speed and direction for an offshore oil rig in the North Sea. Figure 20.1 shows the text produced by SumTime from this data, and also the corpus text for this data (that is, a text written by a human forecaster, which was actually sent to users on the oil rig).

In order to generate the output text shown in Figure 20.1 from the input data shown in Table 20.1, SumTime needs to make many kinds of choices.

- **Choices about document content and structure**: SumTime must decide what information to communicate in the text, and also how the document is structured around this information. This is called *document planning*. In this example, SumTime has decided to communicate information about the wind at the beginning and end of the period, and also at one point in the middle, 15:00 (*'by mid-afternoon'*). The human forecaster has similarly chosen to describe the wind at the beginning and end, and at one intermediate point but has chosen a different intermediate point – 18:00 instead of 15:00.

One of these sentences has a word which is wrong.
Click on it.

- He was walking to the canteen when he slipped on a wet floor.
- I was walking to the canteen when I slipped on a wet floor.
- They was walking to the canteen when they slipped on a wet floor.

Figure 20.2 An example literacy screener question (SkillSum input).

Thank you for doing this.
You got 19 questions right. Click here for more information.
Your skills seem to be OK for your Health and Social Care course.
You got all except 3 of the reading questions right. But you made 5 mistakes on the questions about writing.
Perhaps you would like to take a course to help you with your English.
A course might help you to practise your reading skills, because you said you do not read much.
Click here for Key Skills at XXX College.

Figure 20.3 Example text produced by SkillSum.

- **Choices about linguistic structures**: SumTime must decide which linguistic structures (words, syntax, sentences) should be used to communicate the desired information. This is called *microplanning*. An example of word choice is communicating the time 00:00; in the above example the human forecaster has referred to this time using the phrase *'later,'* while SumTime has used the phrase *'by midnight.'*
- **Choices about word order and forms**: SumTime must decide which forms of words to use, and which order words will appear in, based on the above decisions. This is called *realization*. An example of word-form choice is that both the human forecaster and SumTime use the present participle (*'+ing'*) form of verbs; this is based on the genre conventions of this type of weather forecast.

2.2 *Example NLG system: SkillSum*

SumTime generates short summaries of numerical data for specialist users (workers in the offshore oil industry). Our second example system, SkillSum (Williams & Reiter 2008), generates summaries of performance on an educational assessment (of basic numeracy and literacy skills). Its users are students at further education (community) colleges who are enrolled in a course which requires certain levels of basic skills. The input to SkillSum is some background information about the user, plus his or her responses to a set of multiple-choice questions which test basic literacy or numeracy; an example is shown in Figure 20.2. The output of SkillSum is a short text summarizing the user's performance on the test; an example is shown in Figure 20.3.

Although SkillSum's input data and output text are very different from SumTime's, it must make the same kind of choices. SkillSum's document planner must decide what information to give; for example, how much detail to go into about the user's mistakes. SkillSum's microplanner must decide how to linguistically express this information; for example, whether to say *'You got 19 questions right'* or *'You got 19 questions correct.'* SkillSum's realizer must make low-level word-form and ordering choices, such as using *'questions'* instead of *'question'* in the above sentence.

2.3 Other NLG applications

Many other applications of NLG have been explored, including:

- Summarizing other kinds of data, including medical data (Portet et al., 2007), engineering data (Yu et al., 2007), financial data (Kukich 1983), and sports data (Robin & McKeown 1996). The input to these systems, like SumTime, is usually a combination of numeric data and event records (Reiter 2007).
- Generating initial drafts of documents, such as instruction manuals (Paris et al., 1995), legal documents (Sheremetyeva et al., 1996), clinical documents (Hüske-Kraus 2003), and business letters (Coch 1996). The input to these systems is usually a knowledge base which describes the content of the document; sometimes the NLG system is integrated with the knowledge-base authoring tool (Power et al., 1998).
- Generating explanations of reasoning in AI systems, including expert systems (Lacave & Diez 2004), Bayesian reasoners (Lacave & Diez 2002), and theorem provers (Fiedler 2005). The input to these systems is usually a trace of the reasoning used by the AI reasoning system.
- Generating texts that are intended to persuade or motivate users (Reiter et al., 2003a), make users less anxious (Cawsey et al., 2000), or entertain users (Binstead & Ritchie 1997). The input to these systems is quite varied, but usually includes a user model of the reader.
- Supporting users with disabilities; for example letting blind users examine graphs (Ferres et al., 2006), and helping non-speaking users create stories about what they have done (Reiter et al., 2009).

Despite these efforts, NLG has not been widely used in real-world applications. A number of NLG systems have been used in a limited way; for example SumTime was used for a few years by the Aberdeen office of a weather forecasting company to generate draft forecasts which human forecasters post-edited (Sripada et al., 2004), and indeed an evaluation showed that forecast users preferred some of SumTime's (unedited) forecasts to forecasts written by human meteorologists (Reiter et al., 2005). However, to the best of my knowledge, no NLG system has entered long-term widespread use, in the sense of being used by many organizations for many years. In this regard NLG has been less successful than other areas of NLP such as speech recognition, dialogue systems, machine translation, and

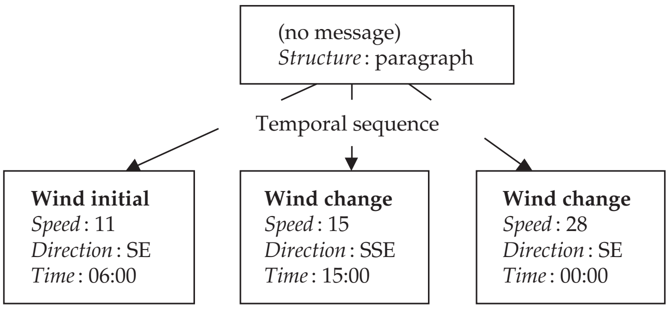

Figure 20.4 Example SumTime document plan.

grammar checking; and also less successful than simple text-generation systems based on fill-in-the-blank templates or mail-merge. My personal belief is that one of the keys to making NLG technology more practically useful on real applications is a better understanding of the research issues which I discuss at the end of this chapter; for example SumTime would probably have been used much more if it had been embedded into an interactive multi-modal meteorological information system (Section 5.3).

3 NLG Choices and Tasks

NLG is often divided into the three stages of document planning, microplanning, and realization. In this section we discuss each of these stages, and briefly summarize the types of choices each stage must make. For more information about the stages, including algorithms and representations, see Reiter and Dale (2000).

3.1 *Document planner*

The document planner decides what information to communicate in the text (*content determination*), and how this information should be organized (*document structuring*).

From a software perspective, the input to the document planner is the input to the entire NLG system; for example numerical weather prediction data in SumTime, and responses to the test questions in SkillSum. The output of the document planner is typically a tree of *messages*. Messages are chunks of information (extracted from the input data), which can be linguistically expressed as a clause or phrase; they are sometimes represented as instances in an AI knowledge-base system such as Protégé.[1] The edges of the tree are often used to represent rhetorical relations between messages. Nodes of the tree can also be annotated to represent document structures (Power et al., 2003) such as paragraphs.

An example of a simple SumTime document plan is shown in Figure 20.4; this is for the SumTime text shown in Figure 20.1. The tree consists of a root node which does not contain a message, but does establish that this text is a 'paragraph' document structure. This node has three children, which are in a temporal sequence relation. Each child communicates a single message about the wind. The first child is a **wind initial** message, which communicates the initial state of the wind; the remaining children are **wind change** messages, which communicate a change in the wind. Each of these messages has the parameters *speed*, *direction*, and *time*. Although SumTime does not use Protégé, these messages could easily be represented in a Protégé ontology (and we would use this representation if we were reimplementing SumTime today).

As this example suggests, the heart of document planning in SumTime is selecting a small number of wind changes to mention, from the input data (Table 20.1). In this case SumTime has chosen to mention the wind changes at 15:00 and 00:00, it assumes the user can interpolate between these points if necessary. This process is guided by a limit to interpolation error, which is based on expert advice from meteorologists as to how much interpolation error is acceptable to end users, considering the tasks that they typically perform using the weather forecasts. Document structure is very simple in SumTime: the messages are always linked by a temporal sequence relation as in the example of Figure 20.4.

As can be seen in this example, document planning typically involves reasoning about the data and how it will be used. Explicit linguistic reasoning may play a role in deciding on document structure (how messages are organized into a tree), but often has a fairly minor role in deciding on document content (which messages are in the tree).

The reasoning performed by document planning can be more complex than SumTime's use of maximum allowable interpolation error. For example, the STOP system (Reiter et al., 2003a), which generated smoking-cessation letters based on the responses to a smoking questionnaire, largely based its document planning on a psychological model of what information should be given to smokers, based on their attitudes towards and beliefs about smoking cessation (Prochaska & diClemente 1992).

Unfortunately, it is difficult to generalize about document planning; for example, the algorithms used by SumTime to decide what information is most appropriate for a weather forecast reader on an offshore oil rig are completely different from the algorithms used by STOP to decide what smoking-cessation information would be most effective for a particular user. In very general terms, the goal of document planning is to identify the information that is useful to the user, and structure it into a coherent document, and it has proven difficult to create a general model of how this should be done.

In the early 1990s Hovy (1993) and other researchers tried to formalize document planning as an AI planning problem based on formal models of rhetorical relations, such as rhetorical structure theory (RST) (Mann & Thompson 1988). However, this approach did not work well in practice. One major problem was that RST-like models of the semantics of relations are too vague to be used in document planners, and also (because they attempt to be general) do not capture important genre-specific aspects of document structure. For example, weather forecasts for offshore oil rigs describe each aspect of the weather (wind, temperature, precipitation, etc.) separately, and describe events about the same aspect (e.g., wind) in strict temporal order. This structure cannot be derived from theories of optimal rhetorical presentation; it is a convention which has arisen in this genre and (for better or for worse) must be adhered to.

An alternative approach to document planning is to try to imitate what human writers do, without explicitly modeling or reasoning about what content would be best for the user. This is typically done by analyzing corpora of human-authored texts and/or explicitly conducting knowledge acquisition sessions with human writers (Reiter et al., 2003b); this approach was used in developing the SkillSum document planner (Williams & Reiter 2005), for example. Recently there has been interest in trying to automate this process using machine learning techniques (Barzilay & Lapata 2005a).

Perhaps the biggest problem with this approach is data sparsity and incompleteness. This kind of analysis usually requires a data-text corpus, which contains the input data (e.g., the data shown in Table 20.1) as well as the human-written texts (e.g., the texts shown in Figure 20.1). Unfortunately, most existing data-text corpora are either too small to provide good coverage of the different cases, and/or do not include all of the data used by the human writers. For example, the SkillSum corpus contained just 16 texts, and Barzilay and Lapata only had corresponding input data for one third of the sentences in their corpus.

Because of these problems, document planners cannot (at the time of writing) be based purely on corpus analysis. Human developers must extrapolate rules to cover gaps in the corpus, based on their domain knowledge and feedback from domain experts and pilot experiments (Williams & Reiter 2005). There is undoubtedly considerable scope for improving the process of creating document planners in this fashion, by improving corpus-analysis and gap-filling methodologies.

3.2 *Microplanning*

The microplanner decides how information is linguistically expressed in the generated text. This process requires many choices to be made, including the following:

- **lexical choice**: choosing which content words should be used to express domain concepts and data;
- **reference**: choosing referring expressions to identify domain entities;
- **syntactic choice**: choosing syntactic structures in generated sentences;
- **aggregation**: choosing how many messages should be expressed in each sentence.

The input to the microplanner is the document plan, which is created by the document planner. The output of the microplanner is a *text specification*; essentially this is a tree whose internal nodes specify document structure (for example, paragraphs), and whose leaf nodes specify deep syntactic structures of sentences.

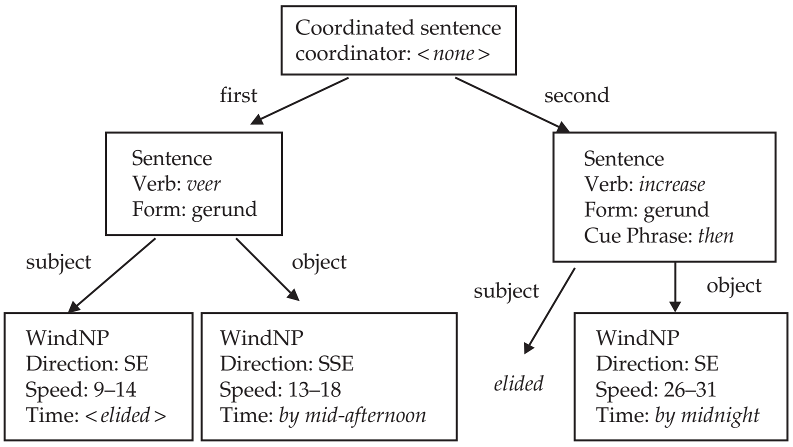

Figure 20.5 Example SumTime deep syntactic structure.

The latter specify content words, coreference, syntactic relationships, and sentence boundaries. The exact form of the deep syntactic structure varies considerably depending on the system's realizer and the grammatical theory (if any) that the realizer is based on.

An example of a deep syntactic structure is shown in Figure 20.5; this is for the document plan shown in Figure 20.4. We do not show the text specification for this example, it would simply contain a root node indicating document structure (very similar to the root node of the document plan shown in Figure 20.4), and a leaf node containing the abstract syntactic structure shown in Figure 20.5.

The deep syntactic structure shown in Figure 20.5 is based on intuitive ideas such as subject and noun phrase, but is customized for the specific genre used in weather forecasts. Thus for example the SumTime realizer accepts a 'syntactic' structure called WindNP which communicates wind speed, direction, and time; this seemed more sensible than trying to force the realizer to use standard noun phrase structures which do not really fit the sublanguage (genre) used in weather forecasts.

In any case, Figure 20.5 illustrates many of the choices that the SumTime microplanner must make. An example of lexical choice is choosing the verb *'veer'* to communicate the change between the wind at 06:00 and the wind at 15:00; this involves deciding which aspect of the change to focus on (in this case the focus is on the change in wind direction), and then choosing the verb based on what occurred in this aspect (in this case *'veer'* as the wind direction changed clockwise). Other lexical choice examples include choosing *'by mid-afternoon'* to communicate the time 15:00, and *'13–18'* to communicate the speed 15. The latter choice could be regarded as a content (document planner) choice instead of a lexical (microplanner) choice, the distinction between the two is not always clear cut.

No referring decisions are needed to generate SumTime texts, including the one shown in Figure 20.5. An example of a referential choice in the SkillSum text shown in Figure 20.3 is *'your Health and Social Care course.'* Other potential ways of referring to this entity include *'your course'* and *'your Health and Social Care course at Aberdeen College'*; the referential choice is choosing which of these referring expressions to use in the generated text.

An example of a syntactic choice in the Figure 20.5 SumTime text is using the gerund form of the verbs (for example, *'veering'* instead of *'veers'*). Another example is the decision to elide (omit) the time phrase from the first WindNP.

An example of an aggregation choice is the decision to express the content in a single sentence. An alternative would have been to use two sentences, for example '*SE 9–14 veering SSE 13–18 by mid-afternoon. Increasing SE 26–31 by midnight.*'

Microplanning choices in SumTime, as in many NLG systems, are made with different levels of sophistication. Some choices are forced by the sublanguage (Grishman & Kittredge 1986); for example SumTime always uses the gerund forms of verbs in wind descriptions, because this is how such texts are conventionally written. Some choices require little computation but are based on substantial linguistic analyses and/or psycholinguistic data. For example lexical choice in SumTime is done using fixed concept-to-word mappings (a trivial computational mechanism), but these mappings are based on an extensive linguistic analysis of word usage in corpus texts and also feedback from users about their word choice preferences. And some choices require non-trivial algorithms (whose design is informed by (psycho)linguistic data); this is the case for aggregation and ellipsis decisions in SumTime.

Perhaps the best studied microplanning choices are referential ones; see Reiter and Dale (2000) for more details on these and other microplanning decisions. In particular a considerable amount of work has been done on the generation of definite noun phrases to identify visible or otherwise salient physical objects (such as '*the red book*'); indeed there has even been a shared task evaluation in this area (Belz & Gatt 2007).

One interesting observation from the work on reference, which has since been observed in other areas of NLG, is that the language produced by human speakers and writers is not necessarily ideal for human listeners and readers (Oberlander 1998). This means that algorithms that generate high-quality referring expressions cannot just be based on corpus analysis and other studies of human language production – they must also be based on studies (often psycholinguistic ones) of human language comprehension.

This raises the more general question of what the goals of microplanning choices are. Is the goal to produce texts that are similar to those produced by human writers (under what metric?); texts that are appropriate for human readers (under what criteria?); or texts that satisfy some other criteria (for example, minimizing legal liability)? Of course, such questions could be asked about all aspects of NLG, but they seem especially prominent in microplanning.

Lexical choice (choosing content words which communicate the information in messages) is extremely important for producing high-quality texts, but is less well understood than referential choices. In principle, it would be nice to base lexical choice algorithms on linguistic theories of lexical semantics (Cruse 1986), but in fact such theories have not proven very useful in NLG. This is because they focus on the relationship between semantic primitives (usually in first-order logic) and words, but very few NLG systems have semantic primitives as their input. The NLG community needs a better understanding of how to map the input data that NLG systems actually get (sensor readings, databases, etc.) on to words (Roy & Reiter 2005), and how this mapping is affected by contextual factors. Such contextual factors include alignment (Brennan & Clark 1996) and other dialogue issues, individual differences between readers' language (idiolect) (Reiter & Sripada 2002), and the desirability or otherwise of varying word choice. These issues are further discussed in Section 5.2. In some cases we also need to consider affective issues (Section 5.4) such as the emotional impact on users of particular word choices; for example, SkillSum needs to use words which not only are understandable and convey the correct meaning, but also are not perceived by the reader as patronizing or disheartening.

A similar situation applies to aggregation choices, that is deciding when and how to combine multiple messages into one sentence. Again aggregation is very important for achieving high-quality texts, but it is not well understood. While many papers have been published on aggregation, there is still considerable uncertainty about when and how to aggregate messages in specific contexts. This is partially because the amount and type of aggregation performed is very dependent on the domain and genre. There are also major differences in the amount of aggregation preferred by different readers, and in the aggregation choices made by different writers.

The final microplanning task we discuss here is high-level syntactic choices, such as whether a sentence should be active or passive, and which tense should be used. In principle many such choices could be based on linguistic theories. For example, centering theory could be used to guide information structure choices such as passivization, and a Reichenbach model could be used to make choices about tenses. In practice such theories often need considerable further elaboration before they are precise enough to be useful in NLG systems (Poesio et al., 2004).

In summary, microplanning requires an NLG system to make decisions about the best way to linguistically express information. With the partial exception of referring expressions, there is little in the way of general theories for guiding NLG microplanners today, instead systems tend to rely on empirical analysis of how language is used in a particular domain and genre. If linguists can help develop a better theoretical underpinning for microplanning tasks, this would be very helpful to the NLG community.

3.3 Realization

The realizer generates an actual text (surface form), based on the information selected by the document planning module and the linguistic choices made by the microplanner. For example, the SumTime realizer generates the text shown in Figure 20.1 from a syntactic structure such as the one shown in Figure 20.5.

Realization is perhaps the best understood aspect of NLG, and indeed many software packages have been developed to carry out this task (which is not true of document planning or microplanning). Most of these packages combine an engine based on a particular grammatical formalism, and one or more grammars based on this engine. These systems include KPML, which is based on systemic functional grammar (Bateman 1997); FUF/SURGE, which is based on functional unification grammar (Elhadad & Robin 1997); RealPro, which is based on meaning-text theory (Lavoie & Rambow 1997); and OpenCCG, which is based on categorial grammar (White et al., 2007). There are also some atheoretical packages which provide less linguistic functionality but allow linguistic constructs to be integrated with templates, such as TG2 (Busemann & Horacek 1998) and Simplenlg (Reiter 2007); van Deemter et al. (2005) discuss in general terms how templates of different levels of sophistication can be used by an NLG system. Morphological generators have also been created, such as Morphg (Minnen et al., 2001).

Some realizers support *overgeneration and selection*. In this mode, the realizer generates several possible surface forms, and uses a separate mechanism to select one of these. The most common selection mechanism is *n*-gram language models which are derived from corpora (these are described in Chapter 3, STATISTICAL LANGUAGE MODELING). OpenCCG, for example, supports this architecture. In principle, an overgenerate-and-select architecture should allow the grammar writing task to be simplified, because subtle constraints such as adjective ordering can be implicit in the language model – they do not need to be explicitly programmed. Also this architecture should make it easier to adapt systems to different genres, because genre linguistic preferences can be implicit in the language model, and do not need to be explicitly coded.

On the other hand, from a practical engineering perspective, it is not clear how to perform testing, quality assurance, and maintenance on systems which are based on language models which are automatically built from corpus texts. For example, if we look at adjective ordering, explicitly encoding adjective ordering rules is a lot of work, but once this is done the rules can be tested and checked, and also modified if users want a different ordering. Implicitly deriving adjective ordering rules from a language model is much less work, but it is harder to test such systems to ensure that they do not produce inappropriate texts in some cases, and more difficult to modify such systems if users request a different ordering.

One pragmatic solution to this problem, which in fact is supported by systems such as OpenCCG, is to combine a symbolic grammar which handles important linguistic decisions which must be correct and which users may wish changed, with a language model which takes care of less important decisions. If it subsequently turns out that a decision made by the language model is more important than initially expected and/or is the subject of user change requests, the symbolic grammar can be expanded to incorporate this decision.

As can be seen from the above discussion, the state of the art in realization is sufficiently advanced (compared to other aspects of NLG) that we can seriously consider software engineering issues such as quality assurance, maintenance, ease of software integration, and documentation quality. Indeed these factors often play a larger role than theoretical factors (such as underlying linguistic theories) when system builders are deciding which realizer to use in their software.

3.4 Other comments

As many people have pointed out (for instance, Mellish et al., 2006), the boundaries between the above stages are not precise. For example, in the above architecture I have said that paragraph boundaries are decided upon in document planning, while sentence boundaries are decided upon in microplanning; but one could also argue that these decisions should be made together, not in different modules. Similarly I have said that the microplanner makes linguistic choices and the realizer implements them; but of course the realizer probably needs to make some choices (especially low-level ones) as well. So the above architecture should be thought of as a starting point; real systems could and should modify it to suit their needs.

The above discussion should also make clear that there are considerable differences in how well different NLG tasks are understood. Our understanding of realization is relatively good, and indeed a number of software packages are available for this task. Our understanding of microplanning is patchy: reasonable in some places (for example, generating definite noun phrases to refer to objects), limited in some places (such as aggregation), and poor in others (such as affective issues in lexical choice). Our understanding of document planning is perhaps weakest of all; current document planners are mostly based on empirical work in a domain with very limited contribution from theoretical models.

4 NLG Evaluation

An important issue in NLG, as in other aspects of NLP, is how to evaluate systems. See Chapter 11, EVALUATION OF NLP SYSTEMS, for a general introduction to the evaluation of natural language processing systems, including terminology, underlying concepts, and common techniques.

In very general terms, one can evaluate how well individual choices are made in NLG; how well NLG modules work; and/or how well complete NLG systems work. Evaluations can also try to determine how effective generated texts are for human readers (reader-based evaluation), or how successfully generated texts match texts produced by human writers (writer-based evaluation). The type of evaluation depends of course on its purpose. For example, if the purpose is to enable a user to decide whether to use an NLG system, then a reader-based system evaluation is most appropriate. On the hand, if the purpose of the evaluation is to test the cognitive plausibility of a model of reference generation, then a writer-based module evaluation would be most appropriate.

Probably the most prominent kinds of evaluation in NLG are reader-based system evaluations, so I will focus on these in the rest of this section. In very general terms, there are three ways of trying to determine how effective an NLG system is at achieving its communicative goal. The most direct approach is an *extrinsic evaluation* which directly measures whether the system achieves its goal. Such evaluations almost always involve human subjects, and generally tend to be relatively expensive and time-consuming. A cheaper alternative is a *non-task-based human evaluation*, where human subjects are asked to read generated texts, and rate, post-edit, or comment on them. A final (and more controversial) alternative is a *metric-based corpus evaluation*; this involves using metrics such as BLEU and ROUGE (these are discussed in Chapter 11, EVALUATION OF NLP SYSTEMS) to measure how similar generated texts are to corpus texts (in other words, we perform a writer-based evaluation in the hope that its results are good predictors of reader-based evaluation).

4.1 Extrinsic evaluations

The most trusted system evaluations are extrinsic ones that directly measure the system's effectiveness at achieving its communicative goal; this is especially true of evaluations which are intended to convince people outside the NLP community, such as doctors, teachers, and mariners. For example, the STOP system was evaluated in a clinical trial with 2,000 smokers (Reiter et al., 2003a). All of the smokers filled out our smoking questionnaire. One third of them received STOP letters, one third received a control non-tailored letter, and one third just received a thank you letter. We contacted them six months after the letters had been sent, and found out how many in each group had managed to stop smoking.

SkillSum was also evaluated extrinsically (Williams & Reiter 2008). We recruited 230 people who were about to start a course at a UK further education college, and asked them to complete the SkillSum literacy and numeracy assessment. The students were again divided into three groups of roughly equal size; two of the groups received variants of SkillSum texts, while the third received a control text (essentially the text produced by the existing software). We asked students to judge whether their skills were sufficient for the course they were signed up for, both before and after they took the assessment and read the texts, and computed how many students in each of the groups increased the accuracy of their self-assessment of their skills.

Both of these evaluations were expensive and time-consuming. The STOP evaluation took 20 months (including planning and data analysis) and cost UK£75,000. The SkillSum evaluation was not separately costed, but required roughly 8 months to carry out (including planning and data analysis) and cost on the order of UK£25,000.

4.2 Non-task-based human evaluations

Extrinsic evaluations may be appropriate as final evaluations of systems with sizable development budgets and timescales, but they may not be realistic for smaller projects and indeed for pilot studies (intended to detect problems and improve systems) in large projects. In such contexts we need evaluation techniques which are quicker and cheaper.

In NLG, such evaluations are usually done using experiments with human subjects, most commonly asking subjects to rate texts on ordinal scales (often Likert scales) or to compare texts; subjects are also usually asked to comment on problems in texts and how to improve them.

For example, the SumTime system was evaluated (Reiter et al., 2005) by generating three alternative texts from data sources: one produced by SUMTIME, one written by humans (professional meteorologists), and one produced by a modified version of SumTime which essentially extracted document plans from the human texts but microplanned and realized these using SumTime. Subjects were shown the texts (without knowing their origin), and asked comprehension questions about the texts; they were also shown different variants of the texts and asked which variant they thought was most accurate, which was easiest to read, and which was overall the best. In another exercise, several systems with SumTime-like functionality were evaluated by having the systems generate alternative forecasts from the same data set, and then asking human subjects to rate the generated texts (Reiter & Belz 2009).

There are a number of other activities we can ask subjects to perform, in addition to rating texts. One is to ask them to edit generated texts and see what changes they make; such an exercise was in fact carried out with SumTime texts (Sripada et al., 2005). Post-editing seems less useful for quantitative comparisons, because there are very large differences in the amount of post-editing which different people perform, and hence a lot of noise in the quantitative data. But it is very useful for qualitative analyses of problems and potential improvements, because the post-edit data tells developers what specific parts of the texts subjects did not like, and how they thought the texts should be improved.

We can also time subjects and see how long it takes them to read texts; this was done in SkillSum, for example, since one of the goals of the project was to generate texts which low-skilled readers could easily read. Simply asking people to silently read a text and press a button when finished is problematic, because some subjects may read in depth while others just skim. It is better to ask subjects to read a text for a concrete purpose, such as answering comprehension questions; another possibility (for low-skill readers in particular) is to ask subjects to read texts aloud. Experiments of both of these types were done in SkillSum (Williams & Reiter 2008).

Last but not least, we can ask subjects to simply read texts and qualitatively comment on them. This can be useful especially in the initial stages of a project; this probably works best with subjects who are articulate and have some idea of the project's goals.

4.3 *Metric-based corpus evaluations*

As discussed in Chapter 11, EVALUATION OF NLP SYSTEMS, some fields of NLP routinely use automatic metrics such as BLEU and ROUGE to evaluate output texts; these metrics work by comparing the output of the system to human-written reference texts. Such metrics are occasionally used in NLG as well, but in a much more limited way than in machine translation and document summarization. This is largely because of doubts as to how well such metrics correlate with (predict) the results of human evaluations.

Reiter and Belz (2009), who worked in the SumTime domain, empirically evaluated how well BLEU and ROUGE scores correlated with human ratings of weather-forecast texts generated by a number of systems. They found that, while BLEU and ROUGE ratings did not correlate with human evaluations of the content quality of generated texts, some variants of BLEU did correlate with human ratings of the linguistic quality of generated texts, provided that the generated texts were produced by systems built with similar technology (BLEU ratings do not correlate well with human ratings of texts produced by systems built with different technologies; Belz & Kow 2009). They conclude that BLEU could be used, with caution, in formative evaluations, but should not be used in summative evaluations (this terminology is explained in Chapter 11, EVALUATION OF NLP SYSTEMS).

Other studies have been more negative. For example, Gatt et al. (2009) evaluated systems that generated referring expressions using extrinsic task-based measures (whether human subjects were able to successfully identify the reference target, and how long it took subjects to do this); non-task-based human ratings, and several automatic metrics (BLEU, ROUGE, and string edit distance). They found that none of the automatic metrics they looked at had a significant correlation with the extrinsic task-performance measures; the human ratings, in contrast, were reasonably well correlated with the results of the extrinsic evaluation.

All of the above studies looked at existing automatic metrics, which were developed for other areas of NLP such as machine translation. Perhaps metrics can be developed specifically for NLG, which do a better job of evaluating NLG systems; this is one of the research challenges for the NLG community.

5 Some NLG Research Topics

In this section I discuss some (by no means all!) current research themes in NLG.

5.1 Statistical approaches to NLG

Statistical corpus-based techniques are very common in other areas of NLP (as is clear from the other chapters in this handbook), and many researchers are investigating how to use such techniques in NLG.

Perhaps the earliest use of statistical corpus-based techniques in NLG was realizers which overgenerated and then used a language model to select between the options (Langkilde & Knight 1998). As mentioned in Section 3.3, this approach is attractive because it reduces the amount of knowledge that must be explicitly encoded in a grammar, although it can raise quality assurance and maintenance concerns.

The overgeneration approach can be used with other selection mechanisms in addition to *n*-gram models. For example, Walker et al. (2007) have created a microplanner based on the overgenerate-and-select approach. The selection is based on ranking rules which are learned from a corpus of example texts which have been ranked by humans. As Walker et al. point out, one advantage of this approach is that it enables the microplanner to adapt to specific genres and even individual differences, if appropriate training data is available; this is very important since dealing with such differences is one of the main challenges in microplanning.

Another statistical approach to NLG is to build statistical models of specific choices that NLG systems must make. Rambow and colleagues have written a number of papers about individual choices, including VP ellipsis (Hardt & Rambow 2001) and lexical choice between near-synonyms (Bangalore & Rambow 2000).

Belz has tried to create a statistical model of the entire microplanning and realization process. Her pCRU framework (Belz 2008) models the generation process as a series of context-free rules. Corpus data is used to create a probabilistic model of the likelihood of each of these rules being used. Given an input data set, the pCRU system then repeatedly invokes the most likely rule, until an output data text is produced; the system also has a Viterbi mode which maximizes the likelihood of all decisions needed to generate a text (the Viterbi algorithm is discussed in Chapter 12, SPEECH RECOGNITION).

The above researchers have tried to create statistical models which can be incorporated into current NLG systems; such models need to be reliable and have good coverage in the target domain. Perhaps for this reason, these researchers have tended to focus on either relatively well-understood choices (such as realization) and/or on domains with relatively simple language (such as weather forecasts). Barzilay and Lapata have taken the different approach of trying to build stand-alone statistical models (i.e., models that are not part of an NLG system) of poorly understood choices such as content selection (Barzilay & Lapata 2005a) and aggregation (Barzilay & Lapata 2006), in linguistically complex texts (sports stories in newspapers). It is not clear if Barzilay and Lapata's models are comprehensive and reliable enough to be used in real systems, but they show how statistical techniques might be used to solve some of the hardest problems in NLG.

Relatively little evaluation has been conducted on how well statistical NLG approaches work when incorporated into complete NLG applications of the kind discussed in this chapter; this is perhaps one reason why the uptake of statistical techniques in the NLG community has been slower than in other NLP communities. One exception is Belz (2008), who created pCRU systems in the SumTime domain, and showed that human evaluators regarded pCRU texts to be similar in quality to human-written weather forecasts.

5.2 *NLG inputs: connecting language to the world*

Of course statistical techniques are an important research focus in almost all areas of NLP. Two research questions which are more specific to NLG concern the inputs and outputs of an NLG system. In this section we will look at some research issues involving inputs; in the next section we will look at some research issues involving outputs.

As mentioned previously, it is rare for the input to an NLG system to be the sort of formal semantic representation which is described in Chapter 15, COMPUTATIONAL SEMANTICS. Usually the input to the system is some combination of databases, knowledge bases, sensor data, event logs, outputs of other systems, and human-authored texts – in other words, the kind of data that computer systems typically store and manipulate. Hence the NLG system must be able to either work with such input data itself, or interface with an external data analysis system. This raises a number of interesting technological, scientific, and even philosophical questions. In this section I will briefly discuss three of the research issues involved in generating texts which are based on non-linguistic input data; see Roy and Reiter (2005) for a description of some of the numerous other challenges involved in connecting an NLP system to real-world input data.

5.2.1 Language and the world: what do words mean? There is a tradition in linguistics, including computational linguistics, of treating language as a 'stand-alone' symbolic system and de-emphasizing how language relates to the non-linguistic world. But of course language evolved to enable humans to communicate about their world (social, intellectual, and physical), not as an isolated symbolic system. In particular, linguistic symbols (that is, words) should have relationship to the non-linguistic world shared by the computer system and its human users.

The problem of mapping data to words is a difficult one, which has received surprisingly little attention in the linguistic community. Lexical semanticists have examined how logical forms are mapped into words, but this is only part of the problem. Using an example from Roy and Reiter (2005), if an NLG system is trying to describe the visual appearance of an object based on camera data, it is not particularly helpful to know that the predicate *Red(X)* maps to the adjective '*red*.' What the system really needs to know is what pixel values from the camera can be described as '*red*,' and how this is influenced by context (lighting conditions, other objects in the scene, user's background, and object being described). The system also may need to know when an object which incorporates many colors can still be described as '*red*' (for example, an apple whose skin is mostly red but has a brown stalk and some greenish areas on its skin). Last but not least, the NLG system may want to know if '*red*' means more than just color; for example, if an apple is visually on the borderline between being '*red*' and '*green*,' perhaps the decision as to which term to use should depend on how ripe it is, since '*green apple*' suggests an apple which is not ripe as well as an apple which is visually green.

Some of these issues were empirically investigated in the SumTime project, especially for choosing phrases to describe time, and choosing verbs to communicate increases and decreases (Reiter et al., 2005). The analysis was performed by aligning the input data (numerical weather predictions) with the corpus texts, and then trying to learn how corpus authors choose words, using both machine learning techniques and manual analysis. The main finding of this investigation was that there were large individual differences in how different forecasters mapped data to words. For example, some forecasters used '*by evening*' to mean roughly 6pm, while others used '*by evening*' to mean midnight. An example involving a verb is that one forecaster said he used the verb '*easing*' (instead of '*decreasing*') to communicate a decrease in wind speed when the absolute wind speed was low; another said he used '*easing*' when the absolute wind speed was high.

There were also differences between the forecast authors and the forecast readers. For example, the forecast authors all used '*later*' to mean near the end of a forecast period, but some forecast users interpreted '*later*' to mean after a period of at least 12 hours. A few of the forecast readers (but none of the authors) also thought that time terms should depend on season (because they were linked to sunrise and sunset) and/or on country (because they were linked to cultural expectations about when people woke up, ate, and went to sleep).

Thus, SumTime showed that there were large differences in how different individuals mapped data to words in the weather domain (and in fact psychologists and linguists have observed such differences in many other domains; Reiter and Sripada 2002). In other words, this aspect of language is not standardized across a linguistic community in the same way that syntax and spelling are. Such problems may be overcome in human language use by lexical alignment mechanisms (Brennan & Clark 1996); these help human participants in a dialogue agree on word usage. However, we do not understand alignment well enough to let a computer NLP system align with a human, and in any case most NLG systems do not participate in dialogues with their users.

5.2.2 Data analysis for linguistic communication A data-to-text system typically must include data analysis and reasoning modules as well as linguistic modules (Reiter 2007). Another interesting research issue is how these modules are affected by the need to generate textual summaries. In other words, how do linguistic constraints and requirements affect the non-linguistic parts of a data-to-text system?

Sripada et al. (2003) addressed this issue, and in particular hypothesized that such data analysis modules are affected by the Gricean maxims (Grice 1975). For example, the Gricean maxim of quality says that utterances should be truthful. One popular data analysis technique is linear segmentation, which involves fitting a straight line to a set of datapoints. For data analysis purposes, this is often done by finding the line which best fits the data points, even if the end points of such a line are not real datapoints. Sripada et al. argue (largely on the basis of linguistic work such as corpus analysis) that if such line segments are explicitly mentioned in generated texts, it is better to use line segments anchored on real datapoints, as this is more truthful.

Sripada et al.'s hypothesis is an interesting one, and more research is needed to investigate to what degree Gricean maxims do indeed affect data analysis and interpretation in systems that produce textual output. If this turns out to be the case, it would be a very interesting example of how a linguistic theory (Gricean maxims) affects what seems to be a non-linguistic task (data interpretation).

5.2.3 Integrating linguistic and non-linguistic knowledge Another important research topic is representations and models that include both linguistic and non-linguistic information. This is not straightforward because a vision system, a knowledge-based reasoner, and an NLP system (for example) may represent very different kinds of information. For example, an NLP system may need to know that *rain* can be communicated using a verb which takes a dummy subject; a knowledge-based meteorological reasoner may need to know that *rain* is a type of precipitation; and a vision system may need to know that *rain* consists of many small semi-spherical objects falling from the sky. It is unclear to what degree it makes sense to try to integrate these types of knowledge into a single representation, and to what degree it is better to keep them distinct. Intuitively it seems that there should be some information which is shared by the different reasoners; for example, *rain* is similar to *snow* from a linguistic perspective (both can appear as verbs with dummy subjects, or as nouns), a knowledge perspective (both are kinds of precipitation), and a visual perspective (both consist of many small objects falling from the sky). But we do not know how to design knowledge representations which capture such commonalities while still effectively representing the information needed by the different kinds of reasoners.

Even without integrated representation in the above sense, systems can still use models and algorithms which utilize both linguistic and non-linguistic data. For example, Kelleher et al. (2005) used both linguistic and visual data in a referring-expression generation task. They computed the visual salience of objects in a visual scene, and used this information, together with a conventional linguistic discourse model, to generate referring expressions to objects in the scene.

5.3 NLG outputs, the role of language in human–computer interaction

From a system perspective, an NLG module is usually part of a larger system whose goal is to inform, assist, motivate, persuade, and/or entertain a user. The larger system may be interactive, include graphical as well as textual outputs, and be sensitive to the user's disabilities, background, and tasks. This raises a number of interesting research questions, ranging from basic questions about the effectiveness of language vs. graphics as a communication tool, to more applied questions such as what sort of language is most appropriate for blind people using a screen reader. Collectively these can be considered as questions about the appropriate role of generated language in human–computer interaction (HCI). As with the language-and-world issues mentioned in the previous section, these issues have not received much attention in the linguistic community.

5.3.1 Text and graphics Perhaps the most studied of these issues is text and graphics. In the 1990s there were many publications about 'media choice,' that is the problem of deciding whether a particular bit of information should be presented linguistically or graphically. For example, Feiner and McKeown (1990) suggested, in the context of generating instruction manuals, that physical location should be communicated graphically, and conditional information should be communicated linguistically. Bernsen (1995) proposed an abstract theory which guided the choice of text vs. graphics. Other researchers pointed out that there were many similarities between text and graphics, and in particular many linguistic phenomena, such as conversational implicature (Marks & Reiter 1990), rhetorical structure theory (André & Rist 1994), and sublanguages (Reiter 1995), applied to graphics as well as text.

Of course, what most users really want is not text-only or graphics-only documents, but rather integrated documents that combine text and graphics. More recent research has focused on integrating text and graphics, primarily in the context of embodied conversational agents (ECA), that is animated characters which move, gesture, and talk. I will not discuss ECAs here as they typically communicate using speech instead of written language, but the interested reader can find out more about recent NLG-related ECA research by looking at proceedings of the Multimodal Output Generation (MOG) workshop (for example, Theune et al., 2007 and van der Sluis et al., 2008). Bateman and his colleagues (Bateman et al., 2001) have looked at integrating text and graphics in written documents, taking into consideration layout issues.

One of the difficulties in this research area is evaluation. Users like graphics, even if they are not actually useful and effective (Law et al., 2005). Hence evaluations of multi-modal systems which are based on user ratings or preferences may not be good predictors of task effectiveness. In this area even more than other areas of NLG, we need careful task-effectiveness studies to evaluate systems, and indeed to provide the basis for good theories of media choice and integration.

One common observation about text and graphics is that the choice depends on user characteristics. Most obviously, graphics are probably not appropriate for a visually impaired user, and speech is not appropriate for a hearing-impaired user. Less obviously but perhaps more importantly, the choice may depend on the expertise of the user. In particular, as many visualization experts have pointed out (Tufte 1983), it is easy to mislead people with graphics, intentionally or unintentionally; viewers need some experience using graphs in order to be able to interpret them correctly. Of course language can also mislead people, but almost all users have decades of experience in using language, whereas relatively few people have this much experience using visualizations.

5.3.2 Interaction Another aspect of the 'HCI of NLG' is interaction. One of the most impressive aspects of current visualization systems is their interactivity; users do not just see a static graph, they can interact with the system and request close-ups, choose to have different information presented, etc. In contrast, most current NLG systems generate static texts which cannot be interacted with; indeed better interactivity is one of the most common requests I receive from potential users of NLG systems. Of course dialogue systems (see Chapter 16, COMPUTATIONAL MODELS OF DIALOGUE) allow interaction, but the current state of the art in dialogue technology makes it difficult to build systems with which the user can reliably interact.

An alternative is to allow the user to interact with an NLG system via mouse clicks and/or keyboard commands, as indeed users interact with visualization systems. One way of structuring this interactivity is via 'dynamic hypertext.' Such NLG systems generate texts which include hyperlinks, and some of these hyperlinks invoke the NLG system to generate different texts. The ILEX system (O'Donnell et al., 2001), for example, generated descriptions of museum items, which had hyperlinks to related items. Clicking on one of these links would invoke ILEX to generate a description of the related item; this description could include references and comparisons to the original description.

Other interaction models are also possible. For example, IGRAPH (Ferres et al., 2006), which helps blind users browse statistical data sets, uses a keyboard interface with keys which allow users to browse forward and backward in the data set, and to get higher-level summaries or lower-level details. This interface is similar to a screen reader, which of course is what most blind users are used to.

Most work on NLG interaction attempts to adapt existing types of interfaces, such as hypertext and screen readers. But perhaps what NLG really needs is new types of interfaces. One interesting idea is WYSIWYM (Power et al., 1998), which uses NLG to help users construct knowledge bases and queries; the user constructs these entities by manipulating a feedback text, which is generated from the actual knowledge base entity or query being authored. Another innovative authoring interface is used in STANDUP (Ritchie et al., 2006), which allows children with learning difficulties to create jokes. STANDUP uses an interface which presents the process of creating a joke as a bus journey, with stops such as 'Word Shop.' Hopefully we will see more experimentation with novel interfaces to NLG systems in the future.

5.3.3 User modeling As should be clear by now, there are major differences between individual users of NLG systems, in terms of the type of language they prefer, (dis)abilities, expertise, and task. In principle it would be nice to represent these differences in a user model, and take the user model into account during the generation process.

Zukerman and Litman (2001) summarize research on user modeling in NLG (and other aspects of NLP). In very broad terms, most research on user modeling for NLG has either explicitly asked users to classify themselves into one of a small number of categories (such as 'novice' and 'expert'), or tried to implicitly acquire user models based on how a user interacts with the NLG system. Neither of these approaches allows very detailed models of users to be created; and hence neither approach allows sophisticated in-depth tailoring of generated texts to the particulars of individual users.

However, the benefit of tailoring texts to shallow user models does not seem to be very high, at least in my experience. Detailed user models probably would allow much better texts to be generated, but we do not know how to acquire such models. Researchers have proposed ideas for acquisition of detailed models; for example perhaps it would be possible to create detailed models (at least of a user's linguistic expertise) via corpus analysis of texts that a user has written, or by asking users to rate a large number of texts and analyzing their ratings (Walker et al., 2007). However, to the best of my knowledge no one has yet tried to use these techniques to obtain a good empirical understanding of how language varies between individuals in a linguistic community. This is a pity, because such research could shed light on fundamental questions about language, as well as improve NLG technology.

5.4 *Affective NLG: going beyond informing and helping users*

Most research on NLG has assumed that the purpose of generated texts is to inform users and help them achieve a task. But of course language is used for many other purposes, including motivation, persuasion, stress reduction, social engagement, and entertainment. NLG systems which have such goals are called *affective* NLG systems (de Rosis & Grasso 2000).

5.4.1 Motivation and persuasion Perhaps the best-studied affective NLG goals are motivation and persuasion. There is of course an extensive literature in psychology and marketing on the best way to motivate and persuade people; and also a rich literature in philosophy and logic in formal models of argumentation. Several NLG systems have attempted to use theories from these communities to persuade and/or motivate people.

For example, the STOP system (Reiter et al., 2003a), which produced tailored smoking-cessation letters, was based on a popular psychological model of how people can change addictive behaviors (Prochaska & diClemente 1992). Unfortunately, an evaluation of STOP showed that the system was not effective; recipients of STOP letters were no more likely to stop smoking than recipients of control non-personalized letters. This may be because STOP used fairly shallow user models, and effective tailoring requires detailed user models (as discussed in Section 5.3.3).

Another approach was taken by Carenini and Moore (2006), who based their system on argumentation and decision theory. Their GEA system generated real-estate descriptions of houses, and tried to make these descriptions more effective (in terms of persuading potential buyers to seriously consider a house) by tailoring the description based on a quantitative model of the buyer's preferences.

GEA was tested in an artificial context (with subjects who were pretending to be house buyers, not real house buyers), with mixed but overall encouraging results.

More recently, Guerini et al. (2007) developed a sophisticated system which attempted to persuade people to visit a museum. Guerini's system was based on an embodied conversational agent, so his system could display facial expressions and use appropriate tones of voice. His system also had rich user models, which included information about cognitive state (beliefs, desires, and intentions), emotional state, and social relationships. Unfortunately Guerini's system has only been evaluated in small pilot evaluations (which did not see a significant effect), so its effectiveness is unclear.

All of the above-mentioned systems relied heavily on user models; it may be that the quality of the user model is one of the main limiting factors in the performance of persuasive NLG systems. As mentioned in Section 5.3.3, acquiring good information about users is a major challenge in itself.

5.4.2 Improving emotional state Another communicative goal is to make people feel better; a particular goal of interest is to reduce stress. We know that stress is often inversely correlated with sense of control; someone who feels they have no control over a situation is likely to feel more stressed than someone who feels that they are at least partially in control. Furthermore we know that, in medical contexts in particular, people are likely to feel more in control if they know more; patients of course are entitled to make decisions about their own health care, but they can only effectively make decisions if they understand their situation. Also, understanding one's medical situation can reduce uncertainty, and reduced uncertainty can reduce stress.

For these reasons, it seems plausible that a system which informs patients about their medical condition could reduce their stress about their medical condition; such a system would be very useful in many medical contexts. This hypothesis was investigated by Cawsey et al. (2000), who built several systems which tried to increase patients' understanding of their medical record, using a browsing dynamic hypertext interface. An evaluation showed that patients liked the system, but unfortunately there was no significant reduction in stress among patients who used the system.

A new area of research is using NLG to promote social interaction, for example by helping friends and relatives provide support to parents of sick babies (Moncur & Reiter 2007), and by helping people with communication disabilities engage in social conversations (Reiter et al., 2009).

5.4.3 Entertainment Last but not least, some NLG systems generate texts that are intended to entertain readers. For example Binstead and Ritchie (1997) developed a computer system which generated puns, and Ritchie et al. (2006) adapted this technology to help humans (children with learning difficulties) create puns. These systems combined theories of what makes a good pun with databases which hold the necessary information about word pronunciation, meaning, etc.

There has also been considerable research on computer systems which generate stories. This includes both research on the content and structure of computer-generated stories (Perez y Perez & Sharples 2004) (much of this is carried out in the computational creativity community), and research on appropriate language for stories (Callaway & Lester 2002). Much of this research is presented at conferences which involve the computer-gaming community, such as Interactive Digital Storytelling (Spierling & Szilas 2008).

6 NLG Resources

I conclude this chapter with a short survey of practical resources which might be of interest to people who want to get involved in NLG research. One good general source of NLG resources is the NLG portal of the Association for Computational Linguistics wiki.[2]

Probably the most commonly requested resource is NLG software. There are a number of realizers (Section 3.3) which are freely available on the web. These realizers vary greatly in linguistic sophistication and ease of use, with the simplest realizers (in linguistic terms) generally being the easiest to use. They also vary in practical aspects such as documentation and programming language.

Unfortunately there are currently no software resources for microplanning and document planning which have been successfully used outside the groups which created them. Hopefully this situation will improve in the near future.

Data resources can also be very useful. Quite a few researchers have made resources available on the web, but few of these have actually been used by other people at the time of writing. Exceptions include the SumTime corpus,[3] which contains numerical weather predictions and human-written forecasts based on these predictions; and the TUNA corpus,[4] which contains scene descriptions and referring expressions produced by human subjects from these scene descriptions.

Turning to information resources, such as textbooks and web sites, contemporary NLP textbooks unfortunately say little about NLG. There is one specialist NLG textbook, Reiter and Dale (2000), but it does not present developments since 2000. Bateman and Zock maintain a useful web page which lists NLG systems, including links to homepages and key references.[5] Other than that, the main source of information is conference proceedings, especially the International NLG Conference and the European NLG workshop; proceedings of many of these are available from the ACL Anthology.[6]

NOTES

1 http://protege.stanford.edu
2 http://aclweb.org/aclwiki
3 www.csd.abdn.ac.uk/research/sumtime/
4 www.csd.abdn.ac.uk/research/tuna/corpus/
5 http://www.nlg-wiki.org/systems/
6 www.aclweb.org/anthology/