Практически значимые сферы применения систем автоматической обработки текстов

Системы ***автоматической обработки текста*** (***АОТ-системы***) по выполняемым функциям (входной и выходной информации) можно классифицировать следующим образом:

	Язык входного текста	Язык выходного текста
1	Естественный-1	Естественный-2
2	Искусственный	Естественный
3	Естественный	Искусственный / Естественный
4	Естественный	Естественный + { Искусственный}

К системам первого типа относятся программы машинного перевода, получающие текст на некотором естественном языке и перерабатывающие его в текст на другом естественном языке. Второй тип – системы генерации (синтеза) текстов по некоторому формальному описанию. Системы третьего типа, наоборот, перерабатывают текст на естественном языке в текст на искусственном (индексирование, извлечение смыслового содержания) или в другой текст на естественном языке (реферирование). К последнему классу отнесем программы, занимающиеся проверкой текста, написанного на естественном языке. Они в результате своей работы либо исправляют входной текст автоматически, либо формируют некоторый протокол замечаний. Пример – система ЛИНАР.

Естественный язык - сложная, многоплановая система, с множеством правил, внутренних связей, имеющая отношение ко всем аспектам деятельности человека. Точность и правильность работы программ определяется глубиной анализа. Достаточно глубокий анализ пока достигается только для определенных узких предметных областей (из-за специфичности подъязыка такой области: в каждой области свои термины, специфические семантические отношения и т.п.).

Для создания систем, работающих со всем естественным языком без потери глубины анализа, в настоящий момент не хватает либо технических возможностей (быстродействия, памяти), либо теоретической базы (например, пока нет даже единой схемы достаточно полного, глубокого и непротиворечивого описания семантики естественного языка). Однако в коммерческих системах, ввиду того, что предназначаются они для большого количества пользователей, разных предметных областей, принята концепция поверхностного анализа, к тому же и производится такой анализ значительно быстрее. Дальнейшее продвижение вперед, использование естественного языка в практических областях невозможно без оснащения этих систем обширными и глубокими (с точки зрения охвата различных явлений языка) описаниями и моделями, созданными лингвистами-профессионалами.

Эта тенденция прогнозируется многими исследователями и прослеживается на примере развития АОТ-систем, уже в наши дни представляющих коммерческий интерес и использующихся при решении следующих прикладных задач:

1. Machine Translation and Translation Aids - машинный перевод;
2. Text Generation - генерация текста;
3. Localization and Internationalization - локализация и интернационализация;
4. Controlled Language - работа на ограниченном языке;
5. Word Processing and Spelling Correction - создание текстовых документов (ввод, редактирование, исправление ошибок)
6. Information Retrieval - информационный поиск и связанные с ним задачи.

Отметим, что это деление несколько условное, и в реальных системах часто встречается объединение функций. Так, для машинного перевода требуется генерация текста, а при исправлении ошибок приходится заниматься поиском вариантов словоформы и т.д.

Машинный перевод

Исторически ***машинный перевод*** является первой попыткой использования компьютеров для решения невычислительных задач (знаменитый Джорджтаунский эксперимент в США в 1954 г.; работы по машинному переводу в СССР, начавшиеся в 1954 г.). Развитие электронной техники, рост объема памяти и производительности компьютеров создавали иллюзию быстрого решения этой задачи. Идея захватила воображение ученых и администраторов. Практическая цель была простой: загрузить в память компьютера максимально возможный словарь и с его помощью из иноязычных текстов получать текст на родном языке в удобочитаемом виде. Однако первоначальная эйфория по поводу того, что столь трудоемкую работу можно поручить ЭВМ, сменилась разочарованием в связи с абсолютной непригодностью получаемых текстов. Приведем в качестве примера результаты работы одной из современных коммерческих систем перевода. Предложим ей перевести народное английское стихотворение, известное нам в переводе «Робин-Бобин» (текст этот очень простой, московские дети изучают его в начальной школе):

Robin, Robin, what a man!	*Малиновка, Малиновка, какой человек!*
He eats as much as no one can.	*Он ест насколько никто не может.*
He ate a lot of fish, he ate a lot of meat.	*Он съел много рыб, он съел много мяс.*
He ate a lot of ice-cream and a sweet.	*Он съел много ледяных-сливки и сладкий.*
He ate a lot of porridge and ten eggs	*Он съел много каша и десять яйцо*
And all the cookies Mother had.	*И вся Мать повары имела.*
He drank a lot of juice, he ate a cake	*Он пил много соков, он съел торт*
Then said: «I have a stomach-ache»	*Затем сказал: «У меня есть желудок- боль»*

Сравним с художественным переводом К.Чуковского:

Робин Бобин Барабек
Скушал сорок человек.
И корову, и быка,
И кривого мясника,
И телегу, и дугу,
И метлу, и кочергу.
Скушал церковь, скушал дом,
И кузницу с кузнецом,
А потом и говорит:
- У меня живот болит!

Следующий пример показывает неустойчивость системы машинного перевода при обработке неоднозначностей. Два предложения по отдельности *Flyer flies*" и *Flyers fly*" переводятся так: *Летчик летает* и *Летчики летают*, если же из тех же словосочетаний составить одно предложение *Flyer flies and flyers fly* получаем *Летчик летает и муха летчиков.*

Конечно, системы, настроенные на определенную предметную область, дают гораздо более приемлемые результаты. Однако в этом случае системы перевода получаются очень узко ориентированными, и попытка использовать их даже в смежных предметных областях дает совершенно непредсказуемые результаты. Подобные эксперименты даже распространены среди любителей пошутить: инструкция по эксплуатации манипулятора-мыши, переведенная с английского языка на русский системой автоматического перевода, использующей специализированный медицинский словарь, превращается в описание всевозможных издевательств над несчастным маленьким грызуном.

Возникают эти проблемы из-за принципиально разных подходов к переводу человека и машины. Квалифицированный переводчик понимает смысл текста и **пересказывает** его на другом языке словами и стилем, максимально близкими к оригиналу. Для компьютера этот путь выливается в решение двух задач: 1) перевод текста в некоторое внутреннее семантическое представление и 2) генерация по этому представлению текста на другом языке. Поскольку не только не решена сама по себе ни одна из этих задач, а нет даже общепринятой концепции семантического представления текстов, при автоматическом переводе приходится фактически делать "подстрочник", заменяя по отдельности слова одного языка на слова другого и пытаясь после этого придать получившемуся предложению некоторую синтаксическую согласованность. Смысл при этом может быть искажен или безвозвратно утерян.

Более реалистичными являются попытки создать системы ***автоматизированного перевода*** - программы, которые не берут на себя полностью весь перевод, а лишь помогают человеку-переводчику справиться с некоторыми трудностями (Computer Aided Translation). Одним из примеров таких систем является Eurolang Optimizer. Его можно рассматривать как нечто переходное между компьютерным словарем и программой-переводчиком, как некий набор предметно-ориентированных глоссариев, снабженный интерфейсом для удобства переводчика: предлагается несколько вариантов перевода, выделенные разными цветами в зависимости от условий применимости; переводчик может с помощью меню определенным образом настраивать словари для более быстрого и правильного выбора нужного эквивалента.

Подобные программные средства могут помочь в решении проблем, связанных с терминологией и вообще со знаниями переводчика о предметной области: одни и те же слова могут по-разному переводиться в зависимости от того, о каком предмете идет речь.

Автоматически может быть решена проблема согласованности. Понятно, что согласованность важна в рамках одного документа: один и тот же термин, даже если его без потери смысла можно перевести несколькими словосочетаниями, должен переводиться одинаково на протяжении всего документа. Однако еще более важной является согласованность в широком смысле - разработка и применение единой концепции интерпретации одного и того же термина на разных языках (скажем, американский разработчик программного обеспечения может быть недоволен, что термин *dialog box* переводится на итальянский как *finestra* (окно) и как *boite* (коробка, ящик) на французский). Ошибки,

возникающие вследствие нарушения согласованности, являются серьезной проблемой, так как, имея только текст-результат перевода, уже невозможно установить, какие термины в оригинале были одинаковыми, а теперь переведены по-разному (в отличие от орфографических ошибок, которые исправить никогда не поздно).

В последнее время также появляются автоматизированные системы «доперевода» или «перевода изменений». Их возникновение связано с тем, что большинство технических текстов (описания, инструкции) не являются целиком новыми (как и явления, продукты, механизмы и т.п., ими описываемые), а содержат в себе лишь некоторые изменения, связанные, например, с усовершенствованием конструкции. Система «доперевода» извлекает из памяти знакомые предложения, а новые куски предлагает переводчику. Заметим, что такой человеко-машинный способ генерации новых текстов также помогает согласованности в стиле и терминологии при переходе от одной версии к другой.

Развитием систем подобного вида можно считать канадскую (Канада – двуязычная страна, постоянно сталкивающаяся с проблемой перевода на государственном уровне) систему генерации прогнозов погоды **Forecast Generator** (**FOG**). Можно считать, что в ней перевод полностью заменен генерацией текстов. В памяти системы хранится 20 миллионов слов и словосочетаний, связанных с прогнозами погоды, что позволяет генерировать как английский, так и французский вариант непосредственно из базы данных. Конечно, успешная работа этой системы в значительной мере объясняется ограниченной природой текстов: сообщения о погоде являются классическим примером подъязыка. Ограниченность словаря, грамматики и семантики дает возможность достичь отличных результатов сравнительно простыми методами.

Генерация текста

С необходимостью генерации хотя бы простейших фраз разработчики практических систем столкнулись еще на заре их создания. Даже в столь примитивно организованной (в плане дружественности пользовательского интерфейса) среде, как DOS, при попытке сгенерировать стандартное сообщение о количестве скопированных файлов мы сталкиваемся с проблемой построения фразы: в зависимости от этого количества необходимо использовать разные слова (в английской версии *file* в случае одного файла и *files*, если больше; в русской - и того хуже: могут встретиться варианты *файл, файла* и *файлов*, причем правила, в каком случае какой из них использовать, достаточно сложны).

По степени сложности и выразительности существующие методы генерации сообщений принято подразделять на 4 класса (часто используются комбинации методов). Рассмотрим их на примере генерации сообщений о копировании файлов.

1) ***Canned-based methods***

Неизменяющийся шаблон - просто печать строки символов без каких-либо изменений.

Для генерации сообщений создаются таблицы шаблонов, которые будут выдаваться в зависимости от ситуации. В нашем варианте при копировании одного файла будет напечатана первая строка таблицы:

```
1 file copied,
```

а в случае, например, трех - третья:

```
3 files copied
```

2) ***Template-based methods***

Изменяющийся шаблон - бесконтекстная вставка слов в образец-строку (именно этот метод используется в MS-DOS):

Шаблон: `<Число> file(s) copied`

может быть использован для генерации сообщений:

```
0 file(s) copied,
1 file(s) copied,
2 file(s) copied
```

3) *Phrase-based methods*

Контекстная вставка.

В зависимости от вида сообщения (контекста) шаблон может быть несколько изменен. Скажем, система может распознавать, с каким окончанием писать слово *file* в зависимости от их количества.

Шаблон: `<Число> <Определение> <file/files при =1, >1><Глагол: время - прош.>`

может использоваться для генерации сообщений:

```
1 file copied,
2 marked files copied,
2 marked files deleted
```

4) *Feature-based methods*

Синтез сообщения на основе набора свойств (грамматических признаков).

Это наиболее сложный метод, он требует привлечения обширных лингвистических знаний, но, в то же время, он и наиболее привлекателен. Предложение определяется набором характеристик составляющих его слов (например, наличие/отсутствие отрицания, настоящее/прошедшее время) и правилами их сочетаемости.

Шаблон:`<Число><Определение><file/files при =1, >1><Глагол: время - любое>`
позволяет генерировать сообщения:

```
1 file should be copied,
1 file was copied,
2 marked files were copied
```

Понятно, что генерация логически связных, целостных текстов является гораздо более сложной задачей: к правилам построения предложений добавляются правила их сочетаемости, правила развития сюжета, соблюдения стиля и т.п. Ввиду невозможности их полной формализации задачу генерации полноценных художественных текстов можно считать на настоящий момент неразрешимой. Однако для некоторых специализированных технических текстов эти правила строго оговорены некоторыми стандартами, немногочисленны и поэтому поддаются формализации. Примером таких текстов могут служить различные инструкции, техническая документация, тем более задача ее автоматической генерации давно назрела.

На Западе уже давно разработка документации превратилась в особую подотрасль разработки любых достаточно сложных систем (в том числе программного обеспечения). Сопроводительная техническая документация весьма разнообразна: руководство пользователя, руководство для менеджера (администратора) системы, руководство по монтажу (инсталляции) и первичному запуску, руководство по эксплуатации, руководство по интегрированию системы с другими устройствами (программами), проектные материалы и т.д. Однако часто пользователь не получает своевременно и в полном объеме необходимый ему материал, соответствующий используемой им версии системы. Это можно объяснить двумя причинами. Во-первых (субъективная причина), подготовка документации - это дополнительная работа, требующая дополнительного времени и дополнительных навыков (разработчику трудно изложить требуемое на понятном рядовому пользователю языке, остальным же надо сначала детально изучить систему). Во-вторых (объективная причина), документация устаревает по ходу модернизации системы.

Поиски решения этих проблем привели в свое время к появлению новой профессии «технического писателя». Однако понятно, что привлечение дополнительных работников ведет к удорожанию продукта. Поэтому в последние годы появились практические системы, осуществляющие помощь в разработке документации, вплоть до ее автоматической генерации. Форма и содержание документации часто выбирается не столько из соображений удобства и полезности для пользователя, сколько из соображений простоты ее создания.

Документация, как правило, содержит графическую и текстовую части. Графическую часть проще сформировать, однако без текстовой не обойтись: в ней описывается семантика продукта (назначение, технические данные, ограничения, детализация работы в разных режимах). Очевидно, что качественная система должна генерировать текст, правильный с точки зрения грамматики и синтаксиса естественного языка. Поскольку предметная область точно определена, а техническая документация составляется по определенным строго заданным правилам, степень формализации в постановке данной задачи существенно выше, чем в задаче машинного перевода, что позволяет надеяться на более высокие результаты.

Локализация и интернационализация

Для того чтобы иметь успех на международном рынке, программные продукты должны быть локализованы, т.е. приспособлены к культурным и языковым нормам потенциальных покупателей.

Для многих программных приложений локализация может быть сравнительно простой, когда основная программа (алгоритм) изменяется незначительно. Конечно, опции меню, сообщения об ошибках, экранные подсказки и другие текстовые строки, вставленные в программу, должны переводиться, но это не создает особых проблем, если при разработке приложения была предусмотрена возможность локализации. Для решения этой задачи программный код и текст должны быть разделены. По установленному стандарту текстовые строки оформляются в отдельном файле, вызываемом из программы. Таким способом текстовые строки можно переводить, не затрагивая исходный код.

Подобные принципы облегчения локализации возможны не для всех приложений. Системы, в которых естественный язык используется не только для формирования сообщений на экране, но и является предметом деятельности самой системы (например, программы-автокорректоры), поддаются локализации с большим трудом. Здесь могут потребоваться большие специализированные словари и полная переработка алгоритмов. Часто эта задача настолько сложна, что разработчик ею заниматься не может, и проблема локализации приложений является заботой пользователя-носителя языка.

В идеале для нашего многоязычного мира программные средства должны быть интернациональными; пользователь, купив версию программы для некоторого языка, не должен покупать другую версию для другого. Назрела необходимость иметь программные средства, позволяющие автоматически настраивать приложение на заданный язык. Пока мы еще далеки от этой цели, но работы в этой области ведутся с большой интенсивностью, особенно в Европе, где в связи с образованием Европейского Союза возникает необходимость вести дела и документацию на всех официальных и некотором количестве неофициальных языков.

Работа на ограниченном языке

Одним из способов разрешения проблем, связанных с обработкой естественного языка, является упрощение и некоторая формализация самих текстов: использование ограниченного языка (подмножества языка). Под ограниченным понимается упрощенный язык, использующий ограниченный словарь, грамматику, строго определенные несложные синтаксические конструкции. Обычно в нем запрещаются длинные предложения, длинные цепочки существительных (типа *решение проблемы разработки систем перевода на базе представления текста в виде последовательности предложений*...), не используются пассивные и негативные конструкции, вводятся строгие правила использования терминов. Тексты должны соответствовать одному из стандартных стилей или даже быть составлены по определенному шаблону, принятому в для документов подобного рода.

Эти правила не являются современным изобретением: именно их обычно применяют при написании технической документации. Достаточно «древним» примером ограниченного языка является «Бэйсик Инглиш», введенный англичанами для общения с туземным населением в колониях. Неожиданно он оказался полезен и для общения самих туземцев друг с другом: колонизация ввела в их быт множество предметов и понятий, просто не имеющих названий в их родных языках. Забавно, что через много лет при «колонизации» Европы и всего мира англоязычными техническими средствами используются практически те же методы. Например, все специалисты в области компьютерной техники пользуются английскими терминами (*файл, принтер* и т.д.), не пытаясь подыскать эквивалент на родном языке, и мы по-русски говорим *word для windows*, а не *слово для окон*.

Применение ограниченного языка делает документ более понятным, удобным для восприятия, он становится легче для переводчиков, поскольку дает меньше возможностей для неоднозначного толкования: такой документ легче составить автору, не являющемуся носителем языка документа. Правительства, особенно в Европе, начинают вводить стандарты на подготовку документации, нормы, по которым требуется использование ограниченных языков, особенно в международной торговле. В связи с этим возникает потребность автоматизации проверки соответствия текста правилам ограниченного языка, создания систем «перевода» с естественного языка на ограниченный.

Boeing, Caterpillar и несколько других компаний призвали вести всю документацию только на ограниченном языке. Ими разработана система Boeing Simplified English Checker для проверки соответствия текстов различным промышленным стандартам и государственным нормам. На ее базе создается программа Clearcheck, не только контролирующая правильность текста на ограниченном языке, но и исправляющая ошибки.

Некоторые разработчики прогнозируют создание систем с использованием ограниченных языков, в которых полный и корректный перевод документации будет производиться без вмешательства человека.

Создание текстовых документов (ввод, редактирование, исправление ошибок)

Нет необходимости говорить о многообразии систем для подготовки текстовых документов: текстовых редакторов, издательских систем и т.п. Они прочно вошли в нашу жизнь, без них не может обойтись ни один пользователь и ни одна область деятельности. Более того, создание текстовых документов – одна из основных сфер применения персональных компьютеров. Использование текстовых редакторов обусловлено не только тем, что они облегчают работу, но и тем, что в последнее время во многих сферах деятельности введены стандарты на подготовку текстов, основанные на применении определенных редакторов.

В отличие от машинного перевода разработка систем редактирования текстов еще на заре своего развития, в 60-е годы, считалась коммерчески перспективной прикладной областью. В настоящее время рынок перенасыщен подобными системами; среди их создателей существует жесткая конкуренция, поэтому при введении одним из поставщиков каких-либо новых возможностей (например, проверка стиля) остальные вынуждены вводить в свои системы нечто подобное. Одним из первых массовых нововведений стало включение в состав текстового редактора программ проверки правописания и внесения необходимых исправлений – ***автокорректоров***. Чтобы придать своему продукту новые коммерчески перспективные свойства, создатели вынуждены все больше использовать лингвистические знания, применять методы морфологического и синтаксического анализа. На очереди - создание систем, выполняющих функции научного редактора, т.е. осуществляющих литературную и научную правку текстов, способных производить сложное автоматизированное редактирование текстов на естественном языке.

Проверка текста в таких системах может вестись в режиме off-line – когда формируется протокол замечаний по тексту, либо в режиме on-line – когда исправление ошибок ведется по мере их обнаружения (возможно, после получения соответствующего подтверждения от пользователя). При обнаружении ошибки система может предложить вариант ее исправления (при наличии нескольких вариантов – их упорядоченный список). Замечания по тексту также могут носить различный характер. Они могут быть локальными (указывается фрагмент текста с ошибкой) и глобальными (выдается диагностическое сообщение, касающееся всего текста, например: «текст труден для восприятия»).

Поиск информации и связанные с ним задачи

Не вызывает сомнений необходимость автоматизации поиска заданных текстовых фрагментов в текстах на естественном языке.

Однако часто даже при поиске информации другого рода (например, аудио- и видео-) работа на самом деле ведется с описаниями на естественном языке (например, для организации поиска фотографий необходимо снабдить каждую из них набором словесных характеристик типа «*портрет, профиль, полный рост, женщина*», «*пейзаж, лес, осень*» и т.п.).

В последних разработках классических систем поиска текста основное внимание уделяется дополнению их разнообразными средствами текстовой обработки, что приводит к расширению возможностей и облегчению работы для пользователя-непрофессионала.

Применение компьютеров не только ускоряет создание и обработку документов, но и чрезвычайно стимулирует рост их количества и объема. Очень многие пользователи регулярно сталкиваются с необходимостью быстро просматривать большой объем документов и выбирать из них действительно нужные. Эта задача возникает при работе с текстовыми базами данных, с электронной почтой, при поиске в Интернете. Сократить количество просматриваемых документов могут помочь системы автоматического ***рубрицировая***. Поток входных документов эти системы распределяют по небольшому количеству классов (тематических рубрик). При этом могут учитываться как чисто внешние показатели документов (объем, расширение имени соответствующего файла и т.п.), так и их содержательные характеристики (название, фамилия автора, ключевые слова), которые могут позволить отнести текст к той или иной рубрике.

Часто в крупных организациях, особенно государственных, правила делопроизводства предписывают сопровождать каждый документ кратким описанием или набором ключевых слов. Во всех указанных случаях была бы весьма полезна возможность автоматически составлять сжатые описания содержания документов – рефераты.

К сожалению, автоматические методы не настолько совершенны, чтобы создать полноценный реферат путем генерации предложений текста. Однако уже сейчас возможно ***автоматическое реферирование*** – составление более или менее информативных и связных рефератов заданного объема (***квазирефератов***) – путем выбора информативных предложений из исходного текста, а также выделение достаточно представительного списка ключевых слов.

В качестве ключевых слов система может выбирать слова, наиболее часто встречающиеся в тексте (и являющиеся при этом информативными, т.е. не предлоги, союзы и проч.), либо использовать для отбора какие-либо синтактико-семантические признаки (из фрагмента: *Определение. Интегралом ... называется ...* можно заключить, что *интеграл* – ключевое слово).

При реферировании из текста отбираются предложения, в наибольшей степени характеризующие его содержание. Например, предложения, содержащие ключевые слова (чем больше, тем лучше), либо отобранные по некоторым особым признакам. Размер реферата (коэффициент сжатия) или количество ключевых слов задается пользователем. Результатом работы такой системы может являться некоторый новый текстовый документ (реферат или набор ключевых слов) или же данный документ, в котором ключевые слова или наиболее информативные предложения выделены по тексту.