Natural Language Generation

Definition: generate text from non-linguistic input (Data --> Doucuments). Requires knowledge of **domain** and **language**.

Stages:

1. **Document planning**: Content selection, Structure,
2. **Microplanning**: Linguistically express text (sentence, words) lexcical / syntactic choice
3. **Realization**: grammatical details

Prestage: Data analysis: Trends and patterns.

Multimodel NLG:

- Speech output
- Text and visualizations
  - Produce separately, OR
  - Tight integration: E.g. text refers to graphic, OR Graphs has text annotations

Building NLG systems:

- Knowledge and corpus analysis, source from

  Imitate a corpus of human-written texts, Ask domain experts, Experiments with users

- Evaluation

Stage 1: Document Planning

Goals:

1. **Text**: information to communicate (Most important & domain-dependent)
2. **Structure:** structure the information, make a coherent context.

Approachs:

- **Theoretical approach**: deep reasoning based on deep knowledge
- **Pragmatic approach**: write schemes imitating human in corpus
- **Statistical approach**: use learning techniques to learn content rule from a corpus

Choose content

Theoretical approach

Based on in-depth knowledge:

- User (knowledge, task, etc.)
- Context, domain, word

Use AI reasoning engine:

- e.g. logical rules + knowledge base -> new information

Not feasible in practice:

- Lack knowledge about user
- Lack knowledge of context
- Very hard to maintain knowledge base, e.g. new users, new regulations

Statistical Approach

Statistical / learning techniques 统计/深度

- Parse corpus, Align with source data, ML learn content rules / **schemas** / cases

Worth considering if large copora.

Pragmatic approach: Schema

Analyse corpora texts (aligned to data), manually infer content structure rules.

Typically based on imitating human patterns.

- Revised based on user feed back

Specially structure as well as content.

Text Structure

Rhetorical relations: describe how the part of the text are **linked** to each other. 修辞关系

Common Rhetorical relations:

- CONCESSION(although, despite) 让步
- CONTRAST(but, however) 转折
- ELABORATION(usually no cue) 详尽阐述
- EXAMPLE(for example, for instance)
- REASON(because, since)
- SEQUENCE(and, also) 系列

Common step

Repeat process for at least 20-30 (cover possible cases); **Merge rules** and **deal with conflicts** (of different authors); May give **priority** to one author , and **imitate** him.

Creating schemas:

Usually just writen in as code in Java or other standard programming languages.

Creating schemes is an art, no solid methodology (yet)

Problems:

- corpus texts likely to be inconsistent
  - especially if several authors wrote texts.
- Some cases not covered in the corpus

- unsuall cases, boundary cases.

Advanced : User-adaptation

Texts should depended on:

- user's personality
- user's domain knowledge(how much we need to explain)
- user's vocabulary(can we use techical terms in the text)
- user's task(what does he need to know)

Hard to get this information?

Personality and perspectives: Text can communicate perspectives, e.g.

- smoking is killing you
- if you keep on smoking, your health may keep on going worse
- if you stop smoking, your health is likely to improve
- if you stop smoking, you'll feel better

How to choose between these? Depends on personality of reader

- Some people react better to positive messages, others to negative
- Some react better to short direct messages, others want these weakened.
- Hard to predict

Conclusion

Content determination is the first and most important aspect of NLG (what information should we communicate).

Mostly based on imitating what is obseved in human-written texts(using schemas, writen in Java)

Also decide on structure(tree structure, rhetorical relations)

Stage 2: Microplanning

Choosing language to express content.

3 sub-tasks:

- **lexical** choice: which word to use
- **reference**: how to refer objects
- **aggregation**: how/when combine phrases into sentence

Problem: Zillions ways of expressing a message in words, which one?

Approaches

Theoretical

Define what "best" means, making microplanning choices that optimised it.

Problem: Hard in practice because we don't have good models of the effects of choices.

Pragmatic

Imitate corpus: use statistical learning if corpus large enough.

Problem: sometimes corpus texts may not be very good from a microplanning perspective.

3 sub-tasks:

1) Lexica choice

choosing right words/lemmas to express the content of the message.

Issues that affect lexical choice

- Frequency (affects readability)
- Formality
- Focus , expectations
- Technical terms
- Convention

EXAMPLE: Statistical-Based Lexical Choice

NLG systems express information in human language. Systems need to “know” what expressions are most suitable for expressing a given piece of information.

Goal: To develop a staticstical algorithm for **lexical choice for quantitative information,** which can

- detect the relationship between data dimensions(aka. attributies) and words
- does not rely on hand-crafted rules
- predict both when and which words should be used
- one word can refer to multiple dimensions

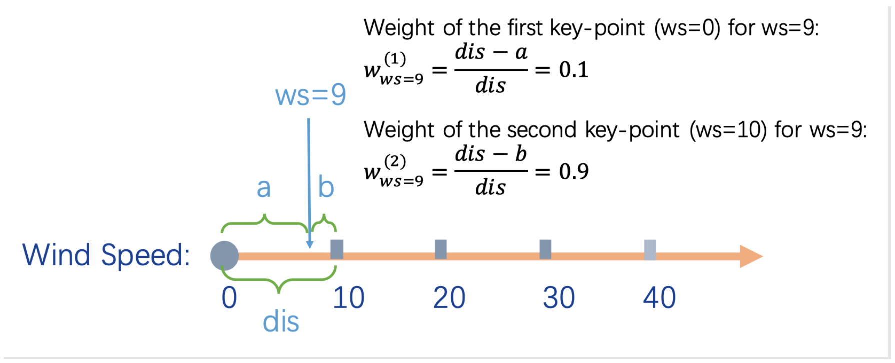
Methodology

Each data record consists of attribute-value pairs.

Representing data in vector

represent each attribute as a combination of some weighted key-points.

- The key-points are derived by
    - Min and max values of attribute(from training data)
    - Key-points are evenly spaced between the min & max
- The number of the key-points for an attribute are fixed



Weight of the first key-point (ws=0) for ws=9:

$$w_{ws=9}^{(1)} = \frac{dis - a}{dis} = 0.1$$

Weight of the second key-point (ws=10) for ws=9:

$$w_{ws=9}^{(2)} = \frac{dis - b}{dis} = 0.9$$

The weight vector of ws = 9 is [0.1, 0.9, 0, 0, 0].

Groups of key-points:

ws=9 --> [0.1, 0.9, 0, 0, 0]

dir = 2 --> [0.97, 0.03, 0, 0, 0]

Concatenate the individual weight vectors:

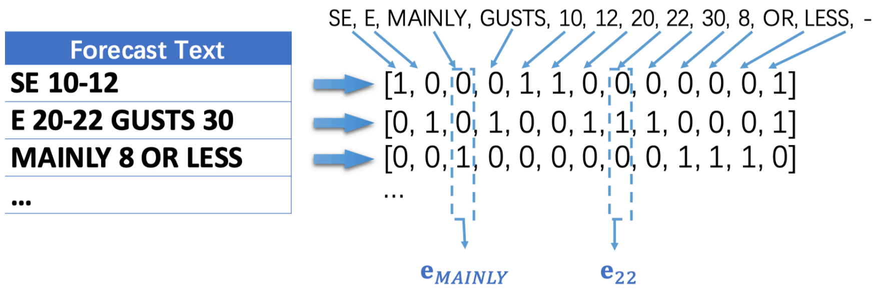
{ws=9,dir=2} -->[0.1, 0.9, 0, 0, 0, 0.97, 0.03, 0, 0, 0]

Entire data-text: vector matrix(**K**), row --> weight vector of a data record.

$$\begin{matrix} \{ws = 9, dir = 2, \dots\} \\ \{ws = 20, dir = 130, \dots\} \\ \{ws = 2, dir = 90, \dots\} \\ \dots \end{matrix} \rightarrow \boldsymbol{K} = \begin{array}{c} \overbrace{\qquad\qquad\qquad\qquad}^{\text{Wind speed}} \; \overbrace{\qquad\qquad\qquad\qquad}^{\text{Wind direction}} \\ \left[\begin{matrix} 0.1 & 0.9 & 0 & 0 & 0 & 0.97 & 0.03 & 0 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & 0 & 0 & 0.66 & 0.44 & 0 & 0 & \dots \\ 0.8 & 0.2 & 0 & 0 & 0 & 1 & 0.86 & 0.14 & 0 & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{matrix}\right] \end{array}$$

Representing word in a vector

Column vector (namely e_i) represent the text of a data record. Each element of e_i indicates whether a word appears in the data record.



现在输入数据和预测文本都用vector表示了。

{ws = 9, dir = 2} $\rightarrow v_{data}$ = [0.1, 0.9, 0, 0, 0, 0.97, 0.03, 0, 0, 0]

“muggy” $\rightarrow V_{muggy}$ = [?, ?, ?, ?, ?, ?, ?, ?, ?, ?]

Methodology

Task: To estimate v_i for word i given a data-to-text corpus as input.

Assumption: v_i and v_d should be close to each other in the vector space if word i appears in data record d

$$\frac{v_{d1} \cdot v_i}{||v_{d1}||\ ||v_i||} = appear(i, d1)$$

$$\frac{v_{d2} \cdot v_i}{||v_{d2}||\ ||v_i||} = appear(i, d2)$$

NB: $appear(i, d1) = 1$ if word i appears in data record d and 0 otherwise

Our task is to find the weight vector v_i for each word i, such that the similarity of v_i and v_d is close to $appear(i, d)$ as much as possible for each data record (d)

$$v_i = \sqrt{\sum_d \left(sim(v_i, v_d) - appear(i, d)\right)^2}$$

Finding v_i equivalent to finding the optimal solution the following equation using least squares.

$$k^{'} \cdot \frac{v_i}{||v_i||} = e_i$$

$$opt(\frac{v_i}{||v_i||}) = (k^{'T} k^{'})^{-1} k^{'T} e_i$$

Once v_i is solved, we can then estimate the most appropriate words for unseen data.

2) Reference

Which phrase should be used to indentify an object?

Referring expression generation: the task of selesting the content(and, to some extent , the form) of referential noun phrases in text.

Types of reference:

- Pronoun - it, them, him, you
- Name - Dr Adam Smith, Adam Smith, Adam, Dr Smith
- Definite NP: the big black dog, the big dog, the black dog, the dog

Suggestion:

- Use pronoun if possible
  - Referent mentioned recently
  - Pronoun is not ambiguous
- Else use name if possible
  - Shortest form which is unambiguous and stylistically allowed
- Else use definite NP
  - Shortest one, prefer basic-level words
- Only use forms seen in corpus

3) Aggregation 聚合

Aggregation: the task of merging distinct representations into a single, more concise representation

Suggestion on Aggregation:

- Generally use the deepest one we can
- Depends on how similar phrases are
- Depends on genre (corpus)

Conclusion

Microplanning

- Decide how to best express a message in language

  Essential for producing “nice” texts
- Imitating corpus works to some degree, but not perfectly

  Currently more of an art than a science
- Key is better understanding of how linguistic choices affected readers:

  Our SumTime weather-forecast generator microplans better than human forecasters

Stage 3: Realisation

Third (last) NLG stage

Creating linear text from (typically) structured input; ensuring syntactic correctness

Take care of details of language

- Syntactic details 句法
- Morphological details 形态
- Presentation details

Problem: There are lots of finicky details of language which most people developing NLG systems don’t want to worry about

Solution: Automate this using a realiser

Syntax

Sentences must obey the rules of English grammar

- Specifies which order words should appear in, extra function words, word forms

Many aspects of grammar are somewhat bizarre

Just tell realiser verb, tense, whether negated, and it will figure out the verb group

- (watch, future) -> will watch
- (watch, past, negated) -> did not watch
- e t c

Similarly automate other “obscure” encodings of informati

Morphology

In linguistics, morphology is the study of words, how they are formed, and their relationship to other words in the same language. E.g.,

- Variations of a root form of a word, e.g., prefixes, suffixes
- Inflectional morphology - same core meaning 变形形态
    - plurals, past tense, superlatives, e.g., dog, dogs
    - part of speech unchanged
- Derivational morphology - change meaning 衍生形态
    - prefix re means do again: reheat, resit s
    - suffix er means one who: teacher, baker
    - part of speech changed

Realiser

Calculates morphological variants automatically:

- (dog, plural) -> dogs;
- (box, plural) -> boxes;
- (child, plural) -> children;
- etc

Automatically insert appropriate punctuation for a structure

Many possible output formats:

- Simple text;
- HTML;
- MS Word

Realiser systems : Examples

simpleNLG – relatively limited functionality, but well documented, fast, easy to use, tested

- Most popular, easy-to-use, programmatically controllable and extendable realisation engine.
- Has adapted into many (western) languages: French, German, Mandarin . . .

KPML – lots of functionality but poorly documented, buggy, slow

openCCG – somewhere in between

many more

Summary

realiser:

- creates linear text from (typically) structured input; ensuring syntactic correctness
- automates the finicky details of language
    - So NLG developer doesn't have to worry about these
    - One of the advantages of NLG