Natural Language Generation

Related terms:

Software Verification, Ontology, Semantic Network, Natural Language Processing, Parsing, Software Testing, Natural Languages, Software Engineering

View all Topics

Deep learning in natural language processing

Chenguang Zhu, in Machine Reading Comprehension, 2021

3.3.1 Network architecture

An NLG model generates text by analyzing the semantics of given text. For example, in an MRC task requiring freestyle answers, the model needs to first analyze the question and article. The NLU models introduced in the previous section can handle this text analysis task. Then, RNN is usually employed to produce text, since RNN can process text with varying lengths, which suits the text generation process of predicting new words given previous context.

To begin with, we prepend <s> and append </s> to all the text. The NLU module converts the input text into a vector with dimension *h_dim*. The RNN module takes the first word <s> and the initial hidden state as the input. RNN then computes the new hidden state , which is used to predict the next word to generate.

Suppose the vocabulary has $|V|$ words (including <s> and </s>). The task of predicting the next word is equivalent to $|V|$-category classification. Therefore we feed into a fully connected layer of size $h\_dim \times |V|$, and obtain a vector with dimension $|V|$. This vector represents the scores given to each word by the model. Thus we select the word with the highest score as the predicted second word.

To continue, the word vector of and the hidden state are fed into RNN to predict the third word. This process goes on until </s> is selected, indicating the end of generation.

Because the above text generation process converts hidden states into words, the corresponding network structure is called a **decoder** (Fig. 3.2). A decoder must use a one-way RNN **from left to right**. If a bidirectional RNN is used, the decoder will peek the words to generate, leading to a nearly 100% training accuracy. However, such a model has no generalization ability.

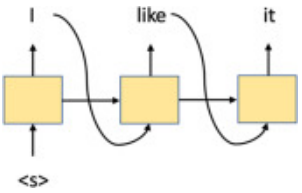

Figure 3.2. An RNN-based decoder for text generation. The first input word is the special symbol <s>.

As the parameters in a neural network are randomly initialized, the decoder will produce text of poor quality in the early stage. Since a generated word is fed into the next RNN module, the generation error will propagate. Therefore a common technique for training the decoder is **teacher forcing**. Under teacher forcing, the word generated by the decoder does not enter the next RNN module during training. Instead, the ground-truth word is used as the next input. This can avoid error propagation and alleviate the cold-start problem, resulting in faster convergence. In practice, one can also intermingle teacher forcing and nonteacher forcing strategy during training. As shown in Table 3.1, in nonteacher forcing, the error starts to propagate from the second generated wrong word *often*, and the subsequent output is completely misguided. During inference, nonteacher forcing is used because the correct answer is unavailable.

Table 3.1. Teacher forcing strategy.

Ground-truth		**<s>**	**I**	**like**	**the**	**weather**	**here**	**</s>**
Teacher forcing	**Input to RNN**	<s>	I	like	the	weather	here	</s>
	Output	I	hate	there	weather	here	</s>	
Non-teacher forcing	**Input to RNN**	<s>	I	often	play	soccer	there	
	Output	I	often	play	soccer	there	</s>	

> Read full chapter

Improving open domain content generation by text mining and alignment

Boris Galitsky, in Artificial Intelligence for Healthcare Applications and Management, 2022

1 Introduction

E-health services are playing an increasingly important role in healthcare management by providing relevant and timely information to patients about their medical care. An important factor in the fast expansion of online health services is the trend in health management towards patient-centric health care, which aims to involve the patient directly in the medical decision-making process by providing better access to the relevant information that patients need to understand their medical conditions and enabling them to make more informed decisions about their prescribed treatment (Dash et al., 2019). Modern physicians are using emerging technologies such as content generation to advance the limits of medical possibilities with new treatments and insights that were once just a dream. At the same time, health systems have never been under such pressure to improve performance, reduce costs, and meet key challenges to safeguard their future (Thimbleby, 2013; Oracle Healthcare, 2021). Content generation in health care allows for the opportunity to provide a personalized, original recommendation to an individual.

Traditional approaches for natural language generation (NLG; McKeown, 1992) rely on three components:

(1) a content planner that selects the data to be expressed

(2) a sentence planner that decides the structure of sentences or paragraphs based on the content plan

(3) a surface realizer that generates the final output based on the sentence plan

Recent studies proposed end-to-end models based on the encoder-decoder framework (Bahdanau et al., 2015) for structured data-to-text generation. Wiseman et al. (2017) employed the encoder-decoder to generate sports game summaries. Mei et al. (2016) proposed an aligner model that integrates the content selection mechanism into the encoder-decoder for generating weather forecasts from a set of database records. Lebret et al. (2016) described a conditional language model for biography summarization. The follow-up studies on biography summarization employed the encoder-decoder framework. These end-to-end models produce fluent text on an ordered input with a fixed structure but have an inferior performance on disordered input. A major problem in open-domain content generation is a

distortion of facts and the truth in a content obtained by sequence-to-sequence models.

Puduppully et al. (2019) proposed Neural Content Planning, which is a two-stage model that includes content planning to handle disordered input. First, the planning uses pointer networks to generate a content plan. Then, the generated content plan is used as the input of the encoder-decoder model such as text generator to generate a description. However, this two-stage model suffers from error propagation between the content planner and the text generator. The generated content plan may contain errors, missing one or more attributes that should be mentioned in the description. As a result, the text generator produces an incomplete description. In an open-domain setting, the encoder-decoder model significantly distorts facts as well, averaging from multiple sources of training data. In this study, we borrow the content planning results from a neural system but improve the truthfulness of generated content by substituting values and phrases from the available content proven to be truthful.

In this chapter, we improve end-to-end content generation with fact-checking and correct fact substitution, to make the content sound and trusted. We refer to the results of neural text generation as **raw** and apply fact-checking to it to identify entities and phrases that are untrue. We then form queries from these untrue phrases and search available sources for sentences to align with the raw ones to retain the syntactic and logical structure and update the values to turn the raw text into the one with correct facts.

1.1 Content generation in health care

Good communication is vital in health care, both among healthcare professionals, and between healthcare professionals and their patients. And well-written documents, describing and/or explaining the information in structured databases, may be easier to comprehend, more edifying, and even more convincing than the structured data, even when presented in tabular or graphic form. Documents may be automatically generated from structured data, using techniques from the field of NLG. These techniques are concerned with how the content, organization, and language used in a document can be dynamically selected, depending on the audience and context. They have been used to generate health education materials, explanations, and critiques in decision support systems (DSS), and medical reports and progress notes (Cawsey et al., 1999).

Creating good content is hard in any vertical, and creating solid content focused around healthcare and medical topics is among the toughest nut to crack. From getting client buy-in to meeting regulatory issues and navigating algorithm updates,

top-notch medical content writing brings its own unique sets of challenges and frustrations (Verblio, 2020).

Different categories of people involved in the healthcare process include consultants, nurses, general practitioners, medical researchers, patients, their relatives, and even accountants and administrators. These people must all be able to obtain and communicate relevant information on patients and their treatment, leveraging the produced content (Rosen et al., 2018). However, there are many obstacles in the way of effective communication; participants may use different terms to describe the same thing. In addition, there is a particular difficulty for patients who do not understand medical terminology. Different participants frequently have distinct information needs and little time to filter information so that no single report is truly adequate for all. As different participants may rarely have time to meet, the care of a patient is shared and passed between them. Hence personalized and thorough quality content is a must.

Writing health-centric hospital content is a headache for many content creators. Creating medical content has a few fundamental challenges, including:

(1) Identifying the expertise required to obtain meaningful expert input.

(2) Conducting all the research and still getting it wrong. In a regulated industry there is a high demand for content correctness.
(3) Supporting the scale. The more challenging the subject matter and scarcity of expertise, the harder the scaling up can be.

1.2 Content generation for personalization

Personalization, that is, adapting to the individual, is becoming an essential component of any computer-based system. In e-health systems, personalization of health information is emerging as a key factor in the trend to patient-centric care. Patient-centric health care aims to engage patients in their treatment to promote greater compliance and satisfaction with their medical treatment, resulting in both better patient outcomes and reduced healthcare costs.

DiMarco et al. (2007) developed a web-based NLG system for the authoring and subsequent personalization of patient education materials. The initial domain is reconstructive breast surgery, but the proposed natural language (NL) software tools and authoring methodologies are generally applicable to all health-related activities and interventions.

Patient educators now support the notion that personalization can:

(1) enhance the uptake of information

(2) bring about increased receptivity to behavioral change

(3) involve patients in their own healthcare decision-making

However, very little is yet known about exactly how to customize language to gain these potential benefits.

Even systems that are designed to provide targeted health information generally do not provide truly personalized content (Galitsky and Kuznetsov, 2013). In a typical case, a patient may have their own "patient portal," which will record their medications, treatments, appointments, and so on; however, the actual content the patient receives will still be generic. In the worst case, the patient might click on a link to gain information about their diagnosed cancer and will receive a pop-up PDF document consisting of a lengthy brochure of "boilerplate" text from a national cancer agency.

Di Marco et al. (2007) combined pre-authoring of input with sophisticated NLG techniques that can automatically edit a text to further refine it. The authors developed a novel paradigm based on generation of new documents from pre-existing text through a process of reuse and revision. The system starts from an existing "Master Document" that contains all the pieces of text that might be needed to tailor the document for any audience. Selections from the Master Document are made according to an individual patient profile, and then are automatically post-edited for personalized data, form, style, and coherence (Fig. 1).

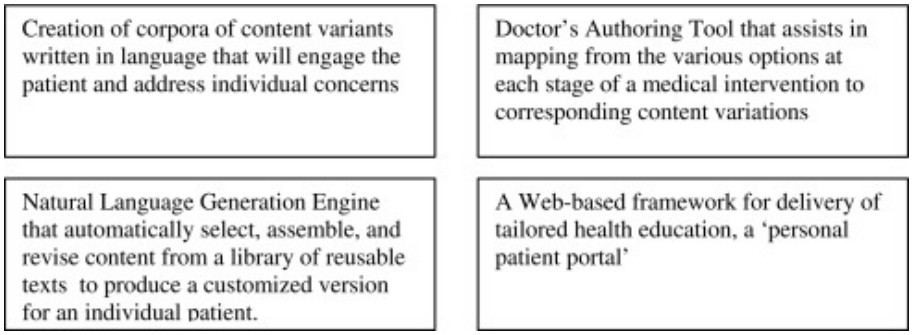

Fig. 1. A high-level view of a personalized content management system for e-health.

Recent approaches to NLG based on external knowledge bases provide good results (Freitag and Roy, 2018). At the same time, most research considers only superficial descriptions of simple pieces of structured data such as attribute-value pairs of fixed or very limited schema, like E2E and WikiBio (Lebret et al., 2016). For real-world, complex databases, it is often more desirable to provide descriptions involving an abstraction and a logical inference of higher generality about database tables and records. For instance, readers should get a kind of a summary over a structured, relational, or NoSQL database.

It should be mentioned that utilizing only data from the table as input for the neural generation model is not sufficient to generate high-quality volume texts. This is triggered by two factors:

(1) Low fidelity and potential diversity. Existing deep learning (DL) approaches, which use only the table as input, have difficulties producing such logically correct generations involving reasoning and symbolic calculations, such as *max*, *min*, *aggregation*, *grouping*, *counting*, and *averaging*.
(2) Uncontrollable content selection. There are various ways to get logically entailed descriptions based on the given table due to a huge number of operations over its values, such as count, comparison, and so forth. A DL model itself cannot reliably select necessary connections due to the difficulty of maintaining high-level semantic constraints in the compositional generation process.

1.3 Natural language generation in intensive care

An implicit assumption of most NLG techniques is that the non-linguistic input information comes from knowledge bases (KBs) with well-defined semantics. In practice, however, in most application domains where automatic textual descriptions are desperately required, such knowledge bases do not exist. The data-intensive clinical environments described in the previous section generate large amounts of clinical data that are not structured into logical forms in a KB. Data-to-text NLG is a recent extension to traditional NLG to allow such naturally occurring data to be described linguistically (Hüske-Kraus, 2003b).

Hallett and Scott (2005) generated summaries of multiple text-based health reports. Cawsey et al. (2000) described a system that dynamically generates hypertext pages that explain treatments, diseases, and more related to the patient's condition using information in the patient's medical record as the basis for the tailoring. Suregen-2 (Hüske-Kraus, 2003a) turned out to be most successful medical data-to-text application that automates the process of writing routine documents, which is regularly used by physicians to create surgical reports. However, the complete summarization of ICU data is a more complex task, involving the processing of time series, discrete events, and short free texts, which has not been accomplished yet.

The system of (Hunter et al., 2008) creates a summary of the clinical data period in four main stages (Fig. 2). All the terms used to describe the discrete events are related to the ontology (Chapter 11). To enable future sharing of this valuable knowledge source, the authors synchronized their ontology with UMLS, a meta-thesaurus that brings together several popular medical ontologies such as SNOMED-CT. The first stage of the processing is *Signal Analysis* component, which extracts the main features of the physiological time series (artifacts, patterns, and trends) using modeling based on a baby's physiological values, auto-regressive filtering, and adaptive bottom-up segmentation techniques.

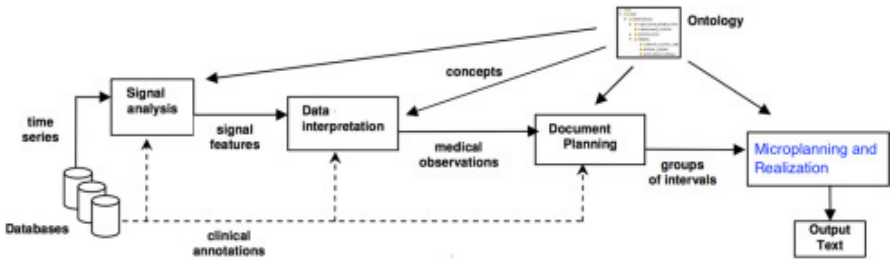

Fig. 2. Content generation in the health domain.

The *Data Interpretation* component performs some temporal and logical reasoning to infer higher medical abstractions and relations ("re-intubation," "A causes B," etc.) from the signal features and the clinical observations using an expert system (Chapter 2) linked to the ontology. From the large number of events generated, the *Document Planning* component selects the most important events based on an importance factor either determined by experts (a surgery must always be included) or computed from the signals (an importance of certain patterns depends on their outcome). The most important events are then structured into a tree. Finally, the *Microplanning* and *Realization* component translates this tree into a text of acceptable readability.

We conclude that in most cases, end-to-end learning can create a high volume of fake, counterfeit content that either needs to be repaired or rejected (Fig. 3).

Fig. 3. Content counterfeiter.

> Read full chapter

Exploiting Natural Language Generation in Scene Interpretation

Carles Fernández, ... Jordi Gonzàlez, in Human-Centric Interfaces for Ambient Intelligence, 2010

4.7.2 Quantitative Results

To retrieve some quantitative measures of the adequacy of the proposed generation, we provide some statistical results that compare the frequencies of use of the two sets of facts: those generated and those collected. The main purpose is to decide up to which point the current capabilities of the vision-tracking and conceptual-reasoning levels enable us to provide natural results.

Figure 4.9 shows statistics about the NL generation experiment. Based on these sorted results, we easily identify facts that should be included, replaced, or removed from the current set. The list of descriptions generated by the system contains many of the most frequent facts: The system generates 100% of those used by more than half of the participants and 77.8% of those employed above the average share of facts (25.9%). The average share was computed as the average of the proportions of the facts in the whole list.

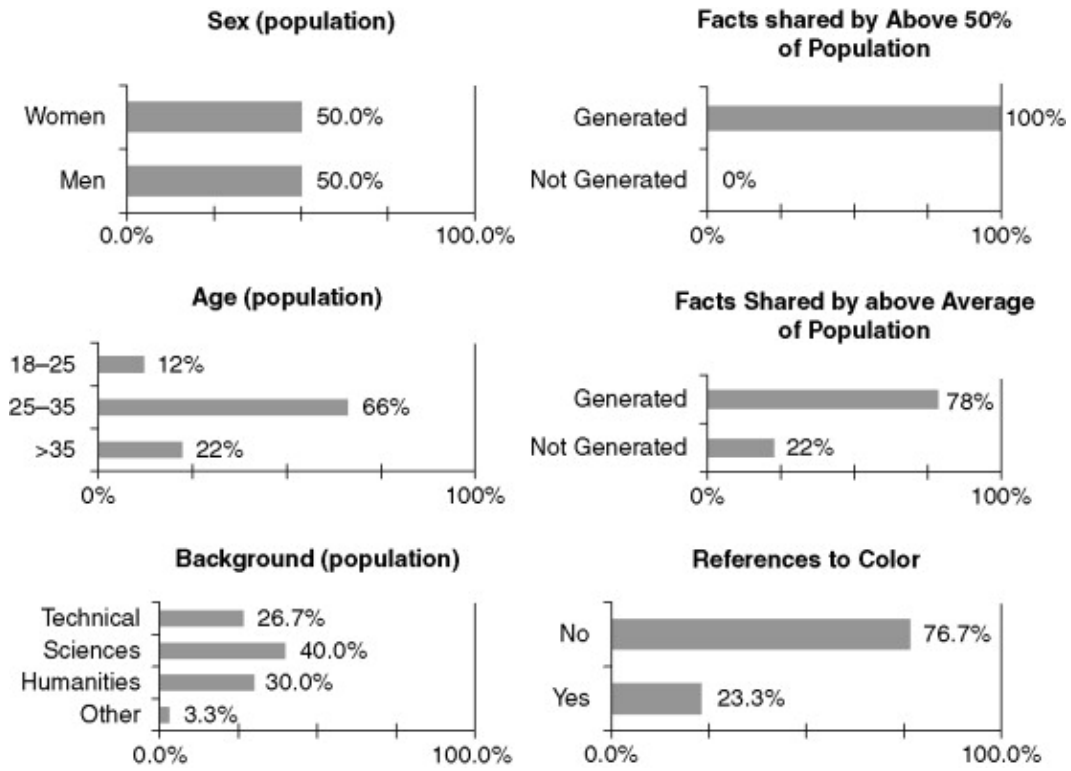

Figure 4.9. Statistics about the NL generation experiment for English and the outdoor sequence. The population consisted of 30 subjects from different backgrounds.

The left column contains information about the population; the right column shows quantitative results about the evaluation and comparison with the facts used.

- First, we notice some of facts referring to the same situations in different ways, such as “danger of runover” and “almost hit/knock down/run over pedestrians” and “pass without stopping/not let pedestrians cross.” Selecting suitable terms depends on the purposed application, and hence it is not identified as a main concern.
- Concerning facts to add, the most significative are those referring to the conversation and interactions between the first two pedestrians (“talk,” “shake hands,” “greet each other,” “wave,” “chat”). This points out an actual limitation of the tracking system, which as yet cannot provide such detailed information about body postures.
- Some of the facts used should be discarded. Some features were detected that seem to indicate that facts are not interesting enough to be included in the final text. These include being obvious (reach the other side after crossing, bend to take an object), being an uncommon expression (having an exchange, motioning someone somewhere), being too subjective (two people being friends), or guessing emotions (seem confused, angry, or surprised). When a situation can be interpreted in more than one way, each interpretation receives less support than a unique one, so that uncertainty is another factor to consider.

It is also interesting that just about one-quarter of the population included color references to support their descriptions. Most of these (above 70%) used a single reference, for the “white car,” which is the only agent with a very distinctive color.

> Read full chapter

Application and techniques of opinion mining

Neha Gupta, Rashmi Agrawal, in Hybrid Computational Intelligence, 2020

1.6.2 Apache OpenNLP

One of the most common tools for NLP is Apache OpenNLP which is based on Java. It provides efficient text-processing services by tokenization, POS tagging, named entity recognition (NER), and many other components used in text mining.

Noteworthy features of OpenNLP are:

1. Natural language generation

2. Summarize
3. NER
4. Tagging (POS)
5. Searching
6. Translation
7. Feedback analysis
8. Information grouping

The Apache OpenNLP library provides classes and interfaces to perform various tasks of NLP. In addition to a library it also provides a CLI, which enhances the functionality. To perform a set of tasks, a predefined model is provided by the OpenNLP which can be downloaded from the website as shown in Fig. 1.14.

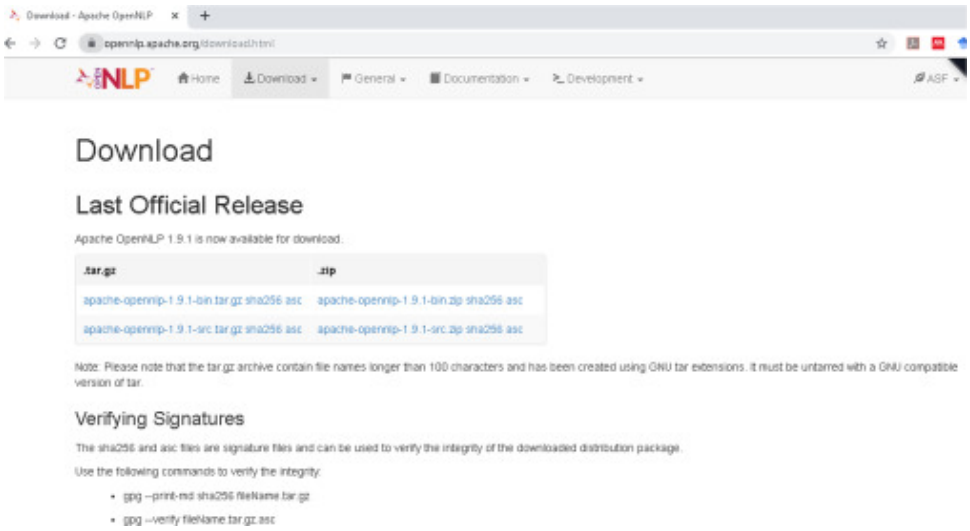

Figure 1.14. OpenNLP.

The various steps of using OpenNLP are:

1. *Sentence detection*: To detect the given text, SentenceModel class is used which belongs to the package *opennlp.tools.sentdetect*. To identify the position of a sentence in the given text sentPosDetect() method is used.For example, “Hello Mohan. Where are you going?” is segmented into two sentences: Hello Mohan and where are you going [6].Sentence detection is a challenging task due to difficulty in identification of the sentence separator as apart from the end of sentence, period (.) can occur in various places in a sentence.
2. *Tokenization*: Tokenization refers to the splitting of a sentence into its smaller parts which are known as tokens. A tokenizer tokenizes a sentence into its root form that is known as the morphological form. For example, “this is my book” is tokenized to “this,” “is,” “my,” “book,” where four tokens are generated from one sentence. Three types of tokenizers available in OpenNLP are *SimpleTo-*

kenizer, *WhitespaceTokenizer*, and *TokenizerME*. These classes contain methods *tokenize()* and *sentPosDetect()*.

3. *Named entity recognition*: In NER the aim is to discover the named things like things, places, and name of the person in a given text. The class for performing his in OpenNLP is *TokenNameFinderModel* class and belongs to package *opennlp.tools.namefind*. Methods used here are *find ()* and *probs()* which are used to detect the names and probability of the sequence, respectively.

4. *Tagging*: Parts of speech tagging was conferred in Ref. [7]. OpenNLP provides the part of speech tags given in Table 1.1Table 1.1. Part of speech tags (POS tag).**POS tagDescription**VBDVerb, past tenseDTDeterminerVBVerb, base formNNP-Proper noun, singularVBZVerb, third person singular presentTOThe word "to"INPreposition or subordinating conjunctionNNNoun, singular or massJJAdjectivePOS tagging can be understood with the following example."Ram had an enemy named Ravan."-a."Ram"—NNP (proper noun);b."had"—VBZ (verb);c."an"—DT (determiner);d."enemy"—NN (noun);e."named"—VBZ (verb);f."Ravan"—NNP (proper noun);g."."—period.

5. *Lemmetization*: After the part of speech tagging we get the tokens of a sentence. In order to further analyze the text, lemmatization is applied. Lemmatization is a technique of finding the base form of a word.For example, in the sentence: "Ram had an enemy named Ravan," the base form of named is name.There are two types of lemmatization in OpenNLP: statistical and dictionary based. In statistical lemmatization a model is built using the training data, whereas in dictionary-based lemmatization a dictionary is required to discover all valid combinations of a word.

6. Parsing: Parsing technique was deliberated in Ref. [8] in which the syntactic structure of a sentence is analyzed using context free grammar. The *ParserModel* and *ParserFactory* class which belong to the package *opennlp.tools.parser* and *ParserTool* class which belongs to package *opennlp.tools.cmdline.parser* are used to parse the sentence.

7. *Chunking*: One of the essential components of chunking is POS information. Chunking can be defined as dividing or grouping the sentence into meaningful word groups, like noun groups and verb groups. The following example illustrates chunking.*"He"—noun phrase"played"—verb phrase"Very well"—Adjective phrase"the teacher"—noun phrase"will teach"—verb phrase"to"—preposition phrase"students"—noun phrase*

> Read full chapter

Brain-like intelligence

Zhongzhi Shi, in Intelligence Science, 2021

14.8.1.1 Natural language processing

Text semantic processing is essentially natural language processing. Natural language processing is not studying natural language as usual but studying a computer system that can efficiently achieve natural language communication, especially soft systems. So it is a part of computer science. Natural language communication between human and computer means that the computer can understand the meaning of natural language text and express the given intention and idea by natural language. The former is called natural language understanding, and the latter is called natural language generation. So natural language processing generally includes natural language understanding and natural language generation.

Natural language processing, that is, natural language communication, or natural language understanding and natural language generation, is very difficult. The root reason is the widespread variable ambiguity in natural language text and dialog. From the format, a Chinese text is a string formed by characters (including punctuation). Characters can form words, words can form sentences, and then some sentences form paragraphs, sections, chapters, and article. Whether it is a variety of levels or a shift from low level to high level, there is the phenomenon of ambiguity. That is, a string with the same format can be understood as different strings under different scenes or context and have different meanings. Under normal circumstances, the majority of these problems can be solved according to the rules of corresponding context and scenes. In other words, there is no overall ambiguity. This is why we do not think natural language is ambiguous, and we can correctly communicate using natural language. On the other hand, as we can see, in order to eliminate it, much knowledge and inference are needed. How to collect and sort out the knowledge completely? How to find a suitable form to save into computer science? How to efficiently use them to eliminate ambiguity? All of them mean very difficult work, and the workload is extremely great. The work cannot be finished by a few people in the short term; it remains a long-term and systematic task.

From recent theory and technology, a universal and high-quality natural language system is also a goal that needs long-term effort. But aiming at certain applications, some practical systems with the ability of natural language processing have emerged. Some of the systems have been commercialized, even industrialized, for example, the natural language interface of database and experts system, all kinds of machine-translation systems, full-text information retrieval systems, automatic abstracting systems, and so on.

> Read full chapter

Big Data Analytics

Venkat N. Gudivada, ... Vijay V. Raghavan, in Handbook of Statistics, 2015

3.6.1 Approaches

Current approaches to automatic summarization fall into two broad categories: *extraction* and *abstraction*. Extractive methods work by selecting a subset of existing words, phrases, or sentences in the original text to form the summary. In contrast, abstractive methods first build an internal semantic representation and then use natural language generation techniques to create a summary. Such a summary might contain words not explicitly present in the original document.

Key challenges in text summarization include topic identification, interpretation, summary generation, and evaluation of the generated summary. Most practical text summarization systems are based on some form of extractive summarization. Abstraction based summarization is inherently more difficult and is an active area of research.

All extraction based summarizers, irrespective of the differences in approaches, perform the following three relatively independent tasks (Nenkova and McKeown, 2011, 2012): (a) capturing key aspects of text and storing it using an intermediate representation, (b) scoring sentences in the text based on that representation, (c) and composing a summary by selecting several sentences.

Open Text Summarizer (OTS) is an open source tool for summarizing texts (Rotem, 2014). It supports over 25 languages, and language-specific information is specified using XML rule files. OTS is available as a library for integration with NLP applications. It is included with many Linux distributions and can be used as a command line tool.

> Read full chapter

Opinion Summarization and Visualization

G. Murray, ... G. Carenini, in Sentiment Analysis in Social Networks, 2017

2.1 Challenges

There are two main challenges associated with summarizing social media text: the documents tend to be noisy and ungrammatical, and they are filled with opinions that may be very diverse and conflicting. Those challenges are addressed in turn.

2.1.1 Challenges of summarizing noisy text

The noisy nature of social media text poses challenges for all summarization systems but the challenges are different for extractive versus abstractive systems.

For an extractive system the output sentences are a subset of the input sentences. This means that the summary will reflect any ungrammaticalities, disfluencies, and slang that are in the input unless care is taken to remove them. This type of postprocessing can include expansion of acronyms, correction of misspellings, appropriate capitalization, and sentence compression. Sentence fragments and ungrammatical sentences are more difficult to deal with; they can be filtered out altogether, at the cost of reducing coverage of the document, or they can be left in the summary as is, at the cost of reducing readability and coherence of the summary.

Abstractive systems do not suffer from the same problem. We can use natural language generation to create well-formed sentences that describe the input documents at a high level. The natural language generation component gives us some control over readability, coherence, conciseness, and vocabulary. However, abstractive systems may be more reliant than extractive systems on syntactic and semantic parsing to represent the meaning of the input sentences at a deeper level. In some domains it is very difficult to get good parsing performance because of the noisy nature of the text. It may be possible to improve parsing by preprocessing the sentences, similar to the extractive postprocessing steps described above. Alternatively, systems could try to incorporate parsers, partial parsers, chunkers, or ontology mappers that have been specifically trained on such noisy data.

In one evaluation and comparison of extractive and abstractive summarizers in the similarly noisy domain of meeting speech [15], user study participants were extremely dissatisfied with extractive summaries and many participants remarked that the extracts did not even constitute summaries. Abstraction seems to have a clear advantage in such domains.

2.1.2 Challenges of summarizing opinion-filled text

The main challenge in summarizing opinion-filled text is to generate a summary that accurately reflects the varied, and potentially conflicting, opinions. If the system input consists of social media posts in which the vast majority of social media users share similar opinions about a person, organization, or product, then simple extractive approaches may suffice: just find some exemplar posts and make those the

summary sentences. However, if the social media users disagree about that entity, or have a similar opinion but for different reasons, extraction alone may not be enough. For instance, if the group exhibits binary polarization about an issue or entity, we may be able to identify a mix of positive and negative exemplar texts, but concatenating them has the potential to create a very incoherent summary.

So abstractive approaches seem well suited to the task of summarizing opinion-filled text. The system can generate high-level sentences that describe the differing opinions and any information or evidence that seems to be driving those opinions. Further, a hybrid system can have each abstractive sentence linked to extracts that exemplify the viewpoint being described.

> Read full chapter

Communication Knowledge

Beverly Park Woolf, in Building Intelligent Interactive Tutors, 2009

5.5.2.1 Basic Principles in Natural Language Processing

NLP addresses issues in formal theories about linguistic knowledge and applied NLP focuses on the practical outcome of modeling human language with the goal of creating software that provides improved human–machine interaction. Researchers in NLP investigate, but are not limited to, the following topics:

- NL *understanding* involves conversion of human language, either input speech (acoustics/phonology) or user typed written words (Figure 5.18, left to right).
- NL generation involves production of natural language from an internal computer representation to either written text or spoken sound (Figure 5.18, right to left). This process often decomposes into three operations: text planning (macroplanning of text content), sentence planning (microplanning of sentence-level organization), and sentence realization (grammatical rendering in linear sentential form).
- *Speech and acoustic input* begins with the understanding of acoustic sound (see Figure 5.18, left box). This includes phonology (the way sounds function within a given language) and *morphology* (the study of the structure of word forms) that address issues of word extraction from a spoken sound or dialogue.
- *Machine translation* involves translation of text from one language to another.
- *Text summarization* involves production of summaries of texts that incorporate the essential information in the text(s), given the readers' interests.
-

Question answering involves responding to user queries, ranging from simple• fact (a single word or phrase) to complex answers (including histories, opinion, etc.).
Discourse analysis involves conversion of human text within a discourse into an internal machine representation, further discussed in Section 5.6.4.

NLP generally focuses on understanding or generating natural language at several levels: *syntax* (the structure of words), *semantics* (the meaning of groups of words), *pragmatics* (the intent of groups of words), and *dialogue* (the exchange of groups of words between people). In generating language, tutors generate phrases, sentences, or dialogue. They might receive a command to perform some communicative act (pragmatics) or create a structure that fixes the prepositional content of the utterance (semantics) that generates a syntactic structure or text or sound. The five phases of NLP suggested in Figure 5.18 provide a convenient metaphor for the computational steps in knowledge-based language processing (the *semantic phase* interprets the student's sentences and the *pragmatic phase* interprets the student's intent). However, they do not correspond directly to stages of processing. In fact, many phases function simultaneously or iteratively and have dual aspects depending on whether the system is understanding or generating natural language. In either case, distinct internal data-structure representations are postulated, and NL systems typically embody mappings from representations at one level to representations at another. A tutor that manages mixed initiative dialogue will both understand students' input speech/text and generate language. It might store all speech input and construct a data structure of phonemes.

Syntax, semantics, and pragmatics impact the correctness of sentences either understood or generated, as the sentences in Figure 5.19 demonstrate.

The following sentences explore the functionality of *syntax, semantics,* and *pragmatics* in forming correct sentences. Suppose your friend invites you to a concert. To understand her intent, you (or an NL processor) must unpack the structure, meaning, and utility of subsequent sentences. Suppose her first sentence is:

"Do you want to come with me to Carnegie Hall?"

Assume the second sentence is one of the following:

Sentence A. "The Cleveland Symphony is performing Beethoven's Symphony No. 5."
*This is a structurally sound sentence and furthers the speaker's intent. It is **correct and understandable**.*

Sentence B. "The ocean water was quite cold yesterday."
*This sentence is structurally correct and semantically sound, but it is unclear how it furthers your friend's intent. It is **pragmatically ill-formed**.*

Sentence C. "Suites have strong underbellies."
*This sentence is structurally correct, but not meaningful. It is **semantically ill-formed**.*

Sentence D. "Heavy concertos carry and slow."
*This sentence is not structurally correct and has unclear meaning. It is **syntactically ill-formed.***

Figure 5.19. Example sentences that explore the role of syntax, semantics, and pragmatics.

- *Sentence A* is structurally sound and furthers the speaker's intent. A listener (human or computer) would easily understand this sentence.
- *Sentence B* is pragmatically ill formed. It does not further the intent of the speaker. Pragmatics addresses the role of an utterance in the broader discourse context.
- *Sentence C* is semantically ill formed based on world knowledge and common sense. It is not meaningful, and the semantic processor would not accept this sentence.
- *Sentence D* is syntactically ill formed. It is not structurally correct, the meaning is unclear, and the syntactic processor would not accept this sentence.

> Read full chapter

Recent trends towards cognitive science: from robots to humanoids

Sarthak Katiyar, Kalpana Katiyar, in Cognitive Computing for Human-Robot Interaction, 2021

The components of a cognitive computing system

The critical component of a cognitive system as shown in Fig. 2.2 are (Hurwitz et al., 2015; Kelly & Hamm, 2013).

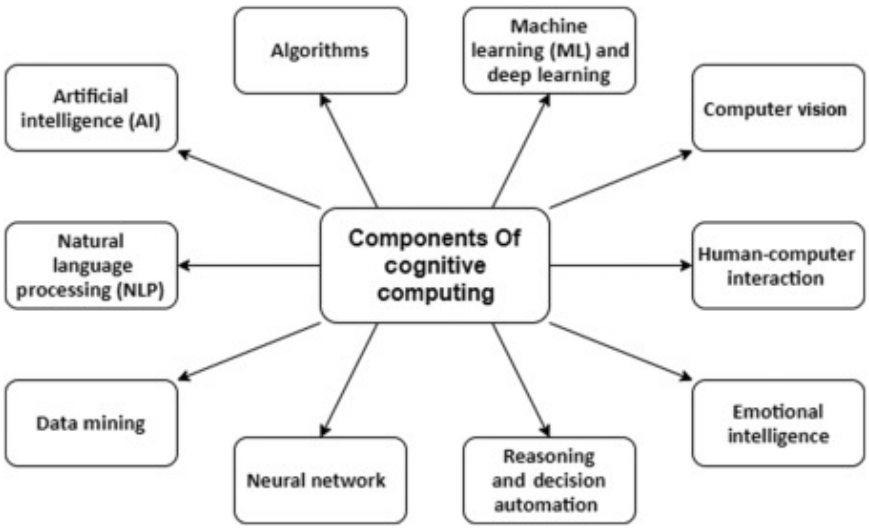

Figure 2.2. Cognitive computing system components.

AI is the expression of intelligence by machines that empower machines to detect and perceive information (input) from the environment for decision-making. Algo-

rithms are like manual books that need to be followed to solve a specific problem by machines. ML is a standard operating procedure for input data processing and making a copy of that future work.

Data mining is the process of implying knowledge and associations of data from large, pre-existing datasets.

Reasoning and decision automation are the part which encompasses learning quality to the CC system to achieve goals. The consequences of the reasoning process reflect in decision automation. That is, the software autonomously generates and implements a solution to a problem.

NLP is computing techniques deployed for response generation that are comprehensible to humans in spoken and written form. It is further subdivided into natural language understanding and natural language generation. Speech recognition authorizes machines to convert speech input into a written language format.

Human–computer interaction (HCI) and dialog and narrative generation: This work as an interface between humans and machines for purposeful and enjoyable communication. This is a subtype of NLP The inclusion of this trait makes CC system more enthralling and upgrades the quality and frequency of interaction among users and machines.

Visual recognition employs high-level algorithms and deep learning to examine individual pictures and patterns of particular images, such as face recognition.

Since the incubation of the CC concept, researchers are focusing on how to integrate the emotional intelligence aspect of humanity in the machine. In this area, a project entitled “MIT startup Affectiva” is launched for building a computing system which recognizes facial movement of humans and respond accordingly.

The neural network is analogous to the human nervous system and consists of nodes/neurons with biased associations. The deep neural network is made up of sheets of neurons. The learning process takes place by upgrading the values of nodes association—the neural network of machine work as a complex decision tree for proposing an answer.

As the practice makes a man perfect, the machine also improves their learning capacity through training, termed deep learning in cognitive system terminology. In this process, different training data set is processed over the neural network of the system, and the result is cross checked with the accurate result given by the living counterpart. In case the result is same, the system has finished its task otherwise the system has to adjust the neural network of its neural interconnections and process the data again. After rigorous training, the neural network learns to generate output that is very similar to human-generated output. The system has now “learned” to perform task as humans do.

Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications

Venkat N. Gudivada, in Handbook of Statistics, 2018

12 Text Summarization

Text summarization is an important NLP task, which has several applications. The two broad categories of approaches to text summarization are *extraction* and *abstraction*. Extractive methods select a subset of existing words, phrases, or sentences in the original text to form a summary. In contrast, abstractive methods first build an internal semantic representation and then use natural language generation techniques to create a summary. Such a summary might contain words that are not explicitly present in the original document. Most text summarization systems are based on some form of extractive summarization.

In general, topic identification, interpretation, summary generation, and evaluation of the generated summary are the key challenges in text summarization. The critical tasks in extraction-based summarization are identifying key phrases in the document and using them to select sentences in the document for inclusion in the summary. In contrast, abstraction-based methods paraphrase sections of the source document.

All extraction-based summarizers perform the following three relatively independent tasks (Nenkova and McKeown, 2011, 2012): (a) capturing key aspects of text and storing as an intermediate representation, (b) scoring sentences in the text based on that representation, (c) and composing a summary by selecting several sentences.

There are two kinds of intermediate representations: topics and indicators. *Topic representations* include a simple table of weighted words. Word weights are based on term frequencies, term frequency × inverse document frequency, and topic words. Other topic representations include using a thesaurus such as WordNet to find topics of semantically related words and assigning them weights; identifying word cooccurrences and assigning weights through latent semantic analysis; and Bayesian topic models which represent document as a mixture of topics with probabilities.

Indicator representations associate a list of indicators such as sentence length, sentence positional information, and occurrence of certain phrases. For example,

LexRank is a graph model which represents a document as a network of interrelated sentences.

For scoring sentences, topic representation approaches compute scores based on how well the sentence expresses some of the important topics in the document or combines the topics. Indicator representations score sentences based on combining evidence from multiple indicators using machine learning techniques. For example, LeXRank assigns weights to sentences by applying stochastic methods to the graph representation of the document.

Approaches for selecting sentences for the summary include *best n*, maximal marginal relevance, and global selection. In the first approach, the top *n* ranked sentences that have the desired summary length are selected. In the second, sentences are selected using an iterative greedy procedure, which recomputes sentence scores after each step. The third approach selects sentences using optimization procedures.

Some recent works in this area include text summarization in the medical domain (Afantenos et al., 2005), empirical methods in text summarization (Das and Martins, 2007), survey of extraction-based text summarization and text mining (Gupta and Lehal, 2009, 2010). In Louis et al. (2010), specific aspects of discourse that provide the strongest indication for text importance is analyzed. They investigate both the graph structure of text provided by discourse relations and the semantic sense of these relations in the context of content selection for single document summarization of news.

It is concluded that structure information is the most robust indicator of salience and semantic sense only provides constraints on content selection. However, sense features help achieve improved performance. Also, the graph structure of text and semantic sense information complement nondiscourse features.

> Read full chapter