UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

INFORMATION ENGINEERING DEPARTMENT

COMPUTER ENGINEERING - AI & ROBOTICS

Natural Language Processing 2023/2024

Francesco Crisci 2076739

Introduction

Natural Language Processing is a field of artificial intelignece that allows machines to read, derive meaning ffrom text, and produce documents. It works in the background of many services, from chatbots through virtual assistants to social media tracking. Such language technologies are already showing major penetration into the information and communication industry.
NLP distinguishes itself from other AI application domains, as for instance computer vision or speech recognition. Text data is fundamentally discrete. But new words can alway be created. Few words are fery frequent, and there is a long tail of rare words. Out-of-Vocabulary words are always being discovered.
Languages have the following characteristics:

- it is **ambiguougs:** units can have different meanings;
- it is **compositional:** meaning of a unit defined as a function of the meaning of its components.
- it is **recursive:** units cna be repeatedly combined.
- they unveils **hidden structure:** local changes in a sentence might have global effects.

Ambiguity

word can belong to several categories, like noun, verbs or modal. The word bank, for example, have different meanings: river bank or money bank. The morphological composition, e.g., the word un-do-able is ambiguous between not doable and can be undone.

Compositionality

At each level, meaning of a larger unit is provided by some function of the meaning of its immediate components and the way they are combined.

Recursion

The rules of the grammar can iterate to generate an infinite number of structures, each with its specific meaning. Recursion is considered the main difference between human and other animals' languages.

Hidden structure

Local changees can disrupt te interprettion of a sentence. This sugests the existence of hidden structure.

Search & Learning

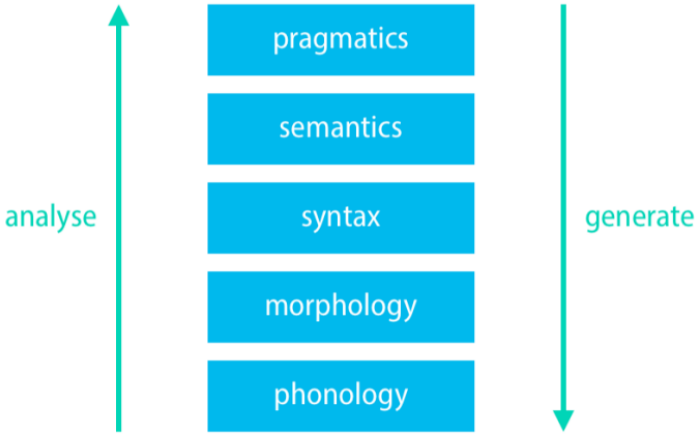
Many natural language processing problems can be written mathematically in the form of optimization:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \in \mathcal{Y}(x)} \Psi(\mathbf{x}, \mathbf{y}; \boldsymbol{\Theta})$$

where x is the input, whicch is an element of a set $\mathcal{X}$; y is the output, which is an element of a set $\mathcal{Y}$(x). Ψ is a scoring function, also called the **model**, which maps from the set $\mathcal{X} \times \mathcal{Y}$ to the real numbers; $\boldsymbol{\Theta}$ is a vector of **parameters** for Ψ. $\hat{y}$ is the predicted output, which is chosen to maximize the scoring function.
The **Search** module is responsible for finding the candidate output $\hat{y}$ with the highest score relative to the input x. the **Learning** module is responsible for finding the model parameters Θ that maximizes the predictive performance.
Structured prediction is an umbrella term for supervised machine learning techniques that involves predicting structured objects, rather than scalar discrete or real values.

Essentials of linguistics

Natural language is a structured system for communication, consisting of a vocabulary and a grammar.
Linguistics is the scientific study of language, and in particular the relationship between language form and language meaning. Besides form and meaning, another important subject of study for linguistics is how languages is used in context. **Phonology** studies the rules that organize pat-



terns of sunds in human languages. It is different from **phonetics**, which is concerned with the production, transmission and perception of sounds, without prir knowledge of the language being spoken.
Morphology is the study of how words are composed by **morphemes**, which are the smallest meaningful units of language. The structure of a word consists of several morphemes: one root or stem and zero or more affixes. **Inflectional morhpology** means that there is no chang ein the grammatical category. **Derivational morphology** means that there is a change in the grammatical category.
A language can be of the following types:

- **Isolating:** a language in which each word form consists typically of a single morpheme;
- **Analytic:** no inflection to indicate grammatical relationship, may still contain derivational morphemes;
- **Synthetic**: uses inflection or agglutination to express relationships within a sentence.

The term **morphologically rich language** refers to a language in which substantial grammatical information is expressed at word level.
what we have seen in our examples so far is **concatenative morhpology:** morphemes are placed one after the other. Some languages, as for instance semitic languages, are based on **template morhpology.**
Syntax studies the rules and constraints that govern how words can be organized into sentences. Differently from formal language theory, NLP systems need to be robust to input that does not follow the rules of grammar.

Part of speech

A **part of speech** is a ccategory for words that play similar roles within the syntactic structure of a sentence. PoS can be defined:

- **distributionally:** Kim saw the {elephant, movie, mountain, error } before we did.
- **functionally:** verbs=predicates; nouns=arguments; adverbs=modify verbs, etc.

Open class tags: noun, verb, adjective, and adverb. New words in the language are usually added to these classes.
Closed class tags: determines, prepositions, conjunctions, etc. Closed word classes rarely receive new members.
There are several representations for syntactic structure. Most common are Phrase structure and Dependency tree.
Phrase structure is a tree-like representation with leaf nodes representing sentence words and internal nodes representing word grouping called **phrases.**
Dependency tree is a tree-like representation where the nodes represent words and punctuation in the sentence, and arc represent grammatical relations between a **head** and a **dependent.** Dependency trees use labels at arcs, representing grammatical relations. Arcs in a dependency tree are often called **dependencies**.
Semantics is the study of the meaning of linguistic expressions such as words, phrases and sentences. The focus is on what expressions conventionally/abstractly mean, rather than on what they migght mean in a particular context. The linguistic study of word meaning is called **lexical semantics.** The internal semantic structure of a word refers to the similarity with other words. The external semantic structure of a word refers to the allowability to combine with other words.
Lexical ambiguity arises because a word can have different meanings, called word senses.
Principle of compositionality: the meaning of a whole expression is a function of the meanings of its parts and of the way they are syntactically combined. Syntax provides the scaffolding for semantic composition: the meaning of a sentence isn't just the amalgamation of the meaning of its component words.
Meaning representation Many representations for semantic structure. Most common are logical, prediccate-argument or graph structure.

Pragmatics studies the way linguistic expressions with their semantic meanings are used for specific communicative goals. In contrast to semantics, pragmatics explicitly asks the question what an expression means in a given context. An important concept in pragmatics is the speech art, which describes an action performed through language.

Discourse analysis

Discourse analysis studies written and spoken language relation to its social context. **Discourse** refers to a piece of text with multiple sub-topics and ccoherence relations between them, such as explanation, elaboration andd contrast. There are many formalisms for representing discourse structure, as for example disccourse representation theory and rhetorical structure theory. **Dialogue** is a cooperative kind of discourse, where two or more participants are involved.

Text Language Processing

Cleaning up a text ddata set requires the use of specialized **regular expressions.** Most programmingg langguages have facilities for compiling regular expressions into efficient finite automata and running these automata on input text. Regular expressions come in many variants. We describe here the so-called **extended** regular expressions. We have different categories and operators:

- **Search:** $/d/$ matches d in woodchuck.
- **Sets of characters:** $/[abc]/$ matches a, b or c; $/[\hat{}abc]/$ matches any character other than a, b and c; $/[a-z]/$ matches any character from a to z; $/./$ matches any character; also called **wildcard**.
- **Aliases:** $\backslash d$ stands for any digit; same as $[0-9]$; $\backslash D$ stands for any character other than a digit, same as $[\hat{}0-9]$; $\backslash w$ stands for any alphanumeric or underscore; $\backslash W$ is the converse of $\backslash w$; $\backslash s$ stands for any whitespace character; $\backslash S$ converse of $\backslash s$; $\backslash.$ stands for period; $\backslash n$ stands for newline.
- **Repetition:** $/\ d\{2,5\}/$ matches between 2 and 5 digits; $/\backslash d\{3,\}/$ matches 3 or more digits; $/\backslash d\{4\}\backslash w?/$ matches exactly 4 digits and one optional alphanumeric or underscore; $/[a-z]+/$ matches one or more lowercase letters; $/\backslash s + java\backslash s + /$ matches java with one or more whitespace characters before and after; $/[\hat{}\backslash+] * /$ matches zero or more characters other than + (Kleen star).
- **Anchors:** $/\hat{}/$ matches beginning of input string or line; $/\$/$ matches end of input string or line; $/\backslash b/$ matches word bondary; $/\hat{}The$ matches occurence of The at the beginning of a string. With | you can have dijunctions and groups $/the|any$ matches the or any; $/grupp(y|ies)/$ matches gruppy and gruppies.
- **Substitute and Back-reference:** we can replace matches of a regular expression by a given pattern using the substitute operator $s/word1/word2$; we can use matches of a regular expression through the back-reference operator.

Words

We need to disstinguish between words and punctuation marks. Punctuation is critical to find sentence boundaries and for identifying some aspects off meaning. There are two ways of talking

about words: **types** are the distinct words appearing in a document and **tokens** that are the individual occurences of words in a document.
the set of all types in a corpus is the **Vocabulary** V. The vocabulary size $|V|$ is the number of types in the corpus. The size of the corpus N is the number of tookens, if we ignore punctuation marks. In very large corpora, the relation between N and $|V|$ can be expressed as

$$|V| = kN^{\beta}$$

In very large corpora, the r-th most frequent type has frequency f(r) that scales acccording to

$$f(r) \propto \frac{1}{(r+\beta)^{\alpha}}$$

The **word-form** is the full infleccted or derived form of the word. Eacch word-form is associated with a single **lemma**, the citation form used in dictionaries. alternatively to $|V|$, another measure of the number of words in a corpus is the number of lemmas.

Corpora

Corpus: large collection of text, in computer-readable form. Several dimensions of varition for corpora should be taken into account: language, genre, etc.
Languag: it is important to tet NLP algorithms on more than one language. languages lacking large corpora are considered low-resource languages.
Genre: text documents might come from newswire, fiction, scientific articles, Wikipedia, etc.
Time: language changes over time. for some languages we have good corpora of texts from different historical periods.
Collection process: how big is the data and how was it sampled? How was the data pre-processed, and what metadata is available?
Annotation: what are the specifics of the used annotation? How was the data nnotated? How where the annotators trained?

Text Normalization

Text normalization is the process of transforming a text into some predefined standard form. This consists of seveeral tasks. There is no all-purpose normalization proccedure: text normalization depends on what type of text is being normalized and what type of NLP task needs to be carried out afterwards. Text normalization is also important for applications other than NLP, such as text mining and WEB search engines.

Language identification

Language identification is the task of detecting the source language for the input text. Several statistical techniques for this task: functional word frequency, N-gram language models, distance measure based on mutual information, etc.

Spell checker

Spell checkers correct grammatical mistakes in text. They use approximate string matching algorithms such as **Levenshtein distance** to find correct spellings.

Contractions

The following should be managed before further normalization: contracted forms, abbreviations and slangs. The most straightforward techniques is to create a dictionary of contractions and abbreviations with their corresponding expansions.

Punctuation

Punctuation marks in text need to be isolated and treated as if they were separatee words. This is critical for finding sentence boundaries and for identifying some aspects of meaning.

Tokenization

Tokenization is the process of segmenting text into units called tokens. Tokenization tecchniques can be grouped into three families: word, character and subword tokenization. Tokens are then organized into a vocabulary and, depending on the specific NLP application, may later be mapped into natural numbers.

Word tokenization

Word tokenization is a very common approach for European languages. For english, most of the text is already tokenized after previous steps, with the following important exceptions: special compound names and city names.

Character tokenization

Major east Asian languages write text without any spaces between words. For most Chinese NLP task, character tokenization works better than word tokenization: each character generally represents a single unit of meaning and word tokenization results in huge vocabulary, with large number of rare words.

Subword tokenization

Many NLP systems need to deal with **unknown words**, that is, words that are not in the vocabulary of the system. To deal with the problem of unknown words, modern tokenizers automatically induce sets of tokens that include tokens smaller than words, called **subwords**. Subword tokenization reduces vocabulary size, and has become the most common tokenization method for large language modelling and neural models in general. Subword tokenization is inspired by algorithms originally developed in information theory as a simple and fast form of data compression alternative to Lempel-Ziv-Welch.
Subword tokenization schemes consist off three diffferent algorithms:

- the **token learner** takes a raw training corpuss and induces a set of tokens, called **vocabulary**.
- the **token segmenter** takes a vvocabulary and a raw sentence, and segments the sentence into the tokens in hte vocabulary.
- the **token merger** takes a token sequence and reconstructs the original sentence.

Three algorithms are widely used for subword tokenization: byte-pair encoding, unigram and WordPiece tokenization.

BPE: learner

The BPE **token learner** is usually run inside words, not merging acoss word boundaries. To this end, use a special end-of-word marker. The algorithm iterates through the following steps:

- begin with a vocabulary composed by all individual characters.
- choose the two symbols A, B that are most frequently adjacent.
- add a new merged symbol AB to the vocabulary.
- replace every adjacent A, B in the corpus with AB.

Stop when the voocabulary reaches size k, a hyperparameter.

BPE: Encoder

Two versions of **BPE token segmenter**: apply merge rules in frequency order all over the data set, or, for each word, left-to-right, match longest token from vocabulary. Encoding is computationally expensive. Many systems use some form of caching: pre-tokenize all the words and save how a word should be tokenized in a dictionary; when an unknown word is seen: applu the encoder to tokenize the word and add the tokenization to the dictionary for future reference.

BPE: Decoder

BPE token merger: to decode, we have to concatenate all the tokens together to get the whole word and use the end-of-word marker to solve possible ambiguities.

WordPiece

WordPiece is a subword tokenization algorithm used by the large language model BERT. Like BPe, WordPiecce starts from the initial alphabet and learns merge rules. The main difference is the way pair A, B is selected to be merged:

$$\frac{f(A,B)}{f(A) \times f(B)}$$

The algorithm prioritizes the merging of pairs where the individual parts are less frequent in the vocabulary.
Text normalization also includes **sentence segmentation:** breaking up a text into individual sentences. This can be undone using cues like periods, question marks, or exclamation points.
Lower casing is very useful to standardize words and before stop word removal. Most programming languages have facilities for string lowercasing.
Stop word removal includes getting rid of common articles, pronouns, prepositions and coordinations. Stop word removal heavily depends on the task at hand, since it can wipe out relevant information.
Stemming refers to the process of slicing a word with the intention of removving affixes. Stemming is problematic in th elinguistic perspective, since it sometimes produces words that are not in the langage, or else words that have a different meaning.
Lemmatization has the objeccctive of reduccing a word to its base form, also called **lemma**, therefore grouping together different forms of the same word. Lemmatization and Stemming are mutually exclusive, and the former is much more resource-intensice than the latter.