1 CL
- Defn: The study of computer systems for underst & gnrt NL.
- Aims: get computers to perform human tasks.
- Appli: sentiment A. Dialogues. ASR.
- Perspectives: [Modelling] Describing contents & format of phenomena description including formatting abstract models/frameworks to understand underlying principle governing lang structures and behaviors. [Methods] Developing tools, procedures, formalism to facilitate NLP by creating algorithms/models. [Tasks] Applying methods to solve specific language tasks.
- Methodologies: [rule-based Approaches] Explicitly model language by defining set of linguistic rules & structures crafted by human experts. Rely on predefined set of linguistic rules. [Statistical app] Implicitly model lang based on patterns and probabilities derived from large datasets, leveraging statil models and ML algorithm to learn. [Sum] lang is rule based yet show stat regularities. Rule based capture explicit linguistic feature, stat reveals implicit patterns. Interplay shows multidimensional nature of CL. - Relation to GL: ① GL is the study of human lang as a universal and recognisable part of human behaviours and cognitive abilities ② common foundation in exploration of lang. ③ Approaches & outlook: GL looks at all langs' universals and principles, CL deals more specifically with technologi parts of lang processing / GL covers entirety of human langs, CL focus on practical applis and the dev. GL - theory foundation. CL - real appli.
- Main subtasks: [NLU] mapping given NL input into useful representns, analysing different aspect of the input: lexicon (pos tagging), morphology (lemmatization), syntax (dependency parsing), discourse (anaphora resolution), pragmatics (sentiment detection). Appli: virtual assistant, sentiment analysis [NLG] producing meaningful NL text from some abstract represtn: subtle (difficult to get it right), very domain-specific.

2. Encoding. Writing Systems: a system to make permanent marks to repre spoken words so it can be rendered almost exactly without involving the speaker. asinEn. Alphabetic: char/letter-sound phoneme, no semantic meaning. abjads: char-all sound, alphabets: char-consonants as in Hebrew. Syllabic: char/symbol-syllable, phonetic sound patterns, x semantic, abgida: families share common consonants but no vowels (Burmese). syllabary: unique symbol for each syllable without systematic organised (日语). Logographic: char/symbol - entire word/morpheme + meaning. Semantic > phonetics (中文). Hybrid combine. Aphabetic chars form syllabic chars tgt. 朝鲜 alphabetic + syllabic + logographic. Emojis are also as do not repre sound-meaning pairings. 2. How lang encode on computers? Info stored in bits, char repre by bit sequences. With 8 bits (1 byte) per char. Encoding systems including ASCII... assign unique code to char to repre text in computers.
3. Binary based 2 symbol (1 & 0). (Big Endian) leftmost - most significant
$\cdots 2^4 2^3 2^2 2^1 2^0$

Decimal to binary: $9 \to 9/2 = 4 \% 1$ ... $4/2 = 2 \% 0$ ... $2/2 = 1 \% 0$ ... $1/2 = 0 \% 1$ 1001
Hexadecimal: 16 symbols.

0	0000	16进从右向左读f 0
1	0001	转2进从右向左读
2	0010	
3	0011	
4	0100	
5	0101	
6	0110	
7	0111	
8	1000	
9	1001	
A 10	1010	
B 11	1011	
C 12	1100	
D 13	1101	
E 14	1110	
F 15	1111	

4. ASCII (American Standard Code for Info Interchange). 7 bits 2^7 chars. For Latin Alphabet (esp EN). Read: 'h' 104 decimal, 68 hexa. ASCII 0x68. Limitations: no umlaunts, no accent / other diacritics, no symbols for currencies other than \$, resulting in different modific across countries, risk of misidentification. 5. Unicode: has a single repre for every char. 8, 16, 32 bits over 1 billion chars. One universal encoding for all lang. Limitations: take up lots of space → three ver.
0... 1110... 1110... 1111... rest to 0补位. * We need encoding systems to ensure compatibility across systems.

3. Writer's Aids / Tech support for writers. 1. Spelling variation across regional differences, different genres.
2. Spelling Error Types. Non-word: string of char that doesn't exist as a word. Reasons: ① Typographical error (know) ② spelling confusion ③ Keyboard layout. Real word: exist, identified by context. Types: long-distance syntax, local syntactic, semantic, repetition. 3. Opaque WS: no direct correspondence between phonological & char repre. Types: silent chara (knights), similar-sounding (sl/th), long/short vowels (receive) double consonant (appliquion). L1-transfer: a person's L1's influence on L2. Error sources.
4. Basic edit operations (Insertion, Deletion, Substitution, Transposition).
5. 3 step create spell checker. Detects errors (source: word lists) → generate candidate corrections (resource: rules, list of similar words). rule-based approach on the list find candidates → ranks: how well fitting the context to suggest correction (Resource: statistical knowledge). 7. Minimal Edit Distance: minimal number of operations required to transfer from nonword to the word. Use DAG (Directed Acyclic Graph) to calculate MED. Why useful: ① Reflect the similarity of contents, ② Being robust against slight misalignment ③ can be efficiently translated into computer programs, modelling after human-typist operations. Nodes: current state of words. Arcs: Action with cost = 1 ↓I →D ↘S.
4. Text as Data. 1. Corpus: structured collection of texts, collected with a specific aim in mind. Usually contains linguistic annotation and metadata. 2. Metadata: Description of non-primary data, e.g. creation data, info of the authorship, tags. 3. Linguistic Annotation is the process of adding linguistic info to text data. incl. POS tags, sentence/word boundaries, parse trees ... 4. TTR (type Token Ratio): assess the diversity of vocab in a text $= \frac{\#\,type}{\#\,tokens}$ number of unique words / number total words. High → high lexical variation (i.e. diversity).

5. Limitations of corpora ① Not fully represented / balanced ② absence / infrequency of a construction 不足 indicate grammatical incorrectness, might due to underrepresentation. ③ No data in result ≠ doesn't exist, might be just not well-repre. Application of corpora. ① Extracting freq. of pos for words ② Estimating n-gram freq ③ Extracting errors in learner language ④ Extract sentiment marker 6. Steps collect data for corpora: Define purpose → Select source → ethical consider → Data collection → preprocessing → annotation → organization → Documentation.

5. Text Classification. 1. How computer learn. The processing of quantifying specific aspects of data, combining them into feature vectors, and using learning algorithms 来 analyse and understand patterns into data. Feature vectors: numerical repre of features of data, a combination of individual features in a dataset. Train, Dev/Validation, Test set.
2. Supervised ML: computer models learn from labelled data (through expert annotations) workflow: split data → train model → test and cross-validate → evaluate. E.g. linear/logistic reg, decision tree, classification. Unsupervised: learning with data without predefined labels. Workflow: define feature extraction → apply algorithm on data set → inspect resulting structure. e.g. K-means clustering, Hierarchy clustering.
3. Evaluate model:

Spam classif	Spam	Ham	
Spam	TP	FN	Recall
Ham	FP	TN	True-Neg Rate (TNR)
	Precision	True Omission Rate	

Precision: $\frac{TP}{TP+FN}$. Recall: $\frac{TP}{TP+FN}$.
Accuracy: $\frac{TN+TP}{TP+FP+TN+FN}$ TNR: $\frac{TN}{FP+TN}$.

4. Tokenization: Breaking down a text into smaller units (字/词/words). Token: an instance of a sequence of chars that're grouped tgt as a useful semantic unit for processing. Challenges: ① contracted/enclitic form: "don't" "he's". ② Hyphenated forms "decile-indept" ③ forms with adjacent to periods. "Str." "U.S.A" ④ Slashes "helpful/fun" "http://" ⑤ special characters: ',,/,!,:?,(,). ⑥ multi-word expression (incl. compound N). "hot dogs" "in spite of". ⑦ Named entities "New York" ⑧ Abbr. "Inc" ⑨ Integrated morphological analysis to split using a lexicon 比如 "USA" "hot dog". Sentence: sequence of words that form a complete grammatical unit and convey distinct idea. Sentence segmentation: decide sentence boundaries (start/end), punctuation chars help to decide. Challenges: ① Different meaning of chars in different language and WS. ② lack of punctuation 和/或 spaces ③ low-resource languages

6. ASR. 1. Speech: acoustic signal produced by human speakers. Phone: basic sound that makes up words, repred with special written symbols in IPA. Phoneme: smallest units of sound that can change meaning of words. The abstract class repreting categories of meaningful sounds in a language. * Prosody (features): syllable stress, intonation, voice quality etc that go beyond individual sounds. Vocal track: physical. Sum: phones are messy real-world realizations in speech, phonemes are abstract labels we used to categorize sounds better.
2. How speech repred with computers? Speech is repreted with computers through a process of digitalisation. Speech as sound wave is captured as an analogue signal through a microphone, then through sampling and quantization these analogue signals are converted into digital data that computers can understand. Computer plays back speech by processing the sequence and using electric current to move those speaker appropriately. Sampling: measuring electrical current from microphone is measured thousands of times per sec and stored as sequences of bits and bytes. Quantization: convert real-valued numbers into integers for computers to repret. 3. Speech processing applications (what task performs, what LT components required). ① Spoken Dialogue System: HCI through spoken language, allow users to communicate with computers via voices | ASR from speech to text | NLU | NLP. ② Speaker Recognition: identify identity from voices | MFCC. ③ Emotion Detection: | MFCC | Sentiment Analysis | Tokenization POS
4. ASR: computer transcribe speech to text. Input: speech signal, output: string of words user said. Challenges/difficulties: ① Ambiguity ② variety in speech (accent, gender, age, L2) ③ Type of speech (context, purpose) read speech, emotional, spontaneous ④ Environmental factors. e.g. background noise, microphone quality ⑤ Single/Multi speaker overlapping turns. Noisy channel Model: analyse audio and find the most likely considering potential noises and errors (assume noisy feature).
ASR system architecture: Audio → Audio Preproc → Decoder → output
transform to feature vectors; lexicon, AcousticM. LM.
Acoustic Model: likely phonemes given audio.
lexicon: mapping written words to corresponding phoneme sequence.
LM: predict prob. rank candidate. Lattice: a huge DAG used to represent the search space. Decoder: navigate through lattice, select most likely based on above. LM's purpose: candidate ranking, models likelihood of word sequence.
Steps in training speech recogniser: data requirement (ideally time aligned) → feature Extraction (MFCC

Mel-Freq-Cepstral-Coefficient) → Training (through lattice, build a statistical models that Decoder will use to explore and rank different path through) → testing (rely on Decoder, Beam search) → evaluation (WER: word Error Rate) → Error analysis (confusion matrix: statistics about which phonemes/words were commonly confused).

7. Text Search · 1. **Why TS challenging**: Due to ambiguity & lack of specificity in queries, resulting in irrevalant e.g. Kate Smith: lack of additional context, shared name. 2 Search Task ① **Searching**: retrieving info that wanted (aka info retrieval) ② **Question answering**: find answer to Q. ③ **browsing**: for music, films, friends, webposts etc didn't even know wanted (recommender system)

3. **Information need**: the info that the searcher is searching for. (A type 的) **intent**: sth that a user wants (to do). **query**: a request/Q posted by users to SE to retrieve specific info 4. General vs specialist users in formulating queries: advanced features including specialized syntax would be powerful to put but opaque to general users, such as Regex. 5. Defn of **Regex**: Strings form a formal language describing patterns of char sequences. **Literals**: char which are identical to what they match (精确匹配对应的字符) e.g. /car/ ⇒ car & "carnival". **Wildcards** 通配符: "." 可匹配任意单个字符(除换行符) e.g. /c.t/ ⇒ cat, cHt, c_t.. **Escaping** 转义: /\$/ = 50$. \d = digits. **Modes** (/g or /i) 全局匹配/g. 不区分大小写/i. **Char sets and ranges**: 匹配[ ]中任一个, 也可用"-"定义字符范围 e.g. /[aeiou]/ 所有元音字母 [a-z] 所有小写字母. **Quantified repetition**: 指定其前面元素出现的次数 "*" 0~n次. "+" 1~n次. "?" 0/1次. 范围 {min, max} e.g. /a{2,3}/ a出现2-3次. **Groups using parentheses** e.g. /(abc)+/ 匹配abc或abcabc.
Anchors /^start/ /end$/. 6. **Document Index**: create index for docs available for searching, allows for efficient and fast retrieval of info, easier to locate specific words, esp with large volumes of texts. **Different approaches**: ① **term by document matrix**: shows which words appear in which doc. terms → rows. docs → columns. 1 in matrix → term appear in. 0 → nowhere. ② **Inverted indices list**: to avoid inefficient ∅s as term-by-doc matrix would be too sparse. Each term is associated with a list of unique doc IDs. Only store IDs of matching docs that IDs, faster processing & more memory efficient. 8. **Evaluating search quality**: ① User Experience survey (happy or not), ② **Objective user interaction logs**. ③ **Precision**: % of docs returned that're relevant (e.g. return 400, 100 rele, Precision = 100/400). **Recall**: % of total rele docs that're returned. (e.g. rele 200, return 200, Recall = 100/400). **F-measure**: combine both: $F_1 = 2 \times \frac{Pre \times Re}{Pre + Re}$ | **Their limits**:

intent: task the user is supposed to achieve with support from assistant. 与 **intent slots/templates** that need to be filled. **intent recognition** to classify user utterance as input. e.g. Siri: [illegible]. **Task-Indep DS**: Design for entertaining/impressing users &/or serve as digital companion. Δ不同实现方式 (implements): ① **Brute force**: hand-written Q/A airs (ELIZA). ② **rule-based**: fix rules to derive response given preceding turns). ③ **corpus-trained chatbox**: based on huge dataset to derive most suitable answer. ④ **language generation**: based on corpus data built ML models, take previous turns as input too. output highest prob of reply (even not existed in corpus) (e.g. ChatGPT) Δ**Evaluation**: **Turing test**: intelligency indistinguishable from that of human. Winograd Schemas: ambiguous pron, need world knowledge. Microsoft's Tay: lack of moral filter. 5. POS Defn: labels assigned to indicate functions. Different tagset & granularity across corpora.

10. LLM. 1. **NN**: interconneted nodes organised in layers. **Transformer**: a type of NN relies on self-attention mechanisms to process data. **what's new about LLM?** ① **unified model** for various tasks, efficient↑. ② **parallelization** (break down task into small & indep & simultly executed) techniques allow for faster training on larger dataset. ③ **attention mechanisms** enhance ability to focus on relevant parts of input. Large in data size, parameter size, model size.

2. **Applications**: Visual Assistance (Siri. Alexa). Language Translation (Google Translate). **Text generation**: (GPT) Chat Bots. Code Gene (GitHub Copilot), **content creation**. 3. **Ethical bias**: not diverse data, social biases, whiteness norms, gender biases). **How social biases incribed in data?** Data collection, human annotators. **Why problematic?** Bias output and decisions when applied LLM in real-world applications. **Fine-tuning**. Due to complexity of language ethical, difficult to ensure only ethical. Try to solve by Reinforcement Learning from human feedback. 4. **Environment Impact**: has negative impact with training consuming significant power and emitting CO_2. Training LLM ~~like GPT-4 can~~ · Carbon intensity varies based on energy sources; coal-powered data centers huge emissions. High computational demands amplify power consumption, leading to environment strain.

Pre- doesn't account for relevant docs that were not retrieved by SE. **Re**- hard to know how many rele docs were missing by SE in real life as we often lack complete datasets or ground truth annotations for evaluation. **F1**- high & most effectiective. Overload, only 1st page result.
✱ **Why ranking is important**: allows SE to prioritize rele. On top result page, Pre > Recall.

8. CALL (Computer-Assisted Language Learning) 1. **L1 characteristics**: ① Divergence between linguistic levels: learner mistakes in different parts of language (sound, word form, sentence structure ...) ② categories for native language not applicable, L2 learning mistaken that native won't make. ③ Difficult to determine target hypotheses: learners making multiple mistakes and show inconsistencies. ④ Often more than 1 error.
NLP in language learning: 1/ **analysing language for learner**: NLP search relevant & appropriate examples/texts. 2/ **analysing learner production**.
2. **ITS** (Intelligent Tutoring System): Δ**Defn**: a computer program helps learner's learning (automatic, immediate feedback, many users 同时使用) Δ**Goals**: 1/ close gap between ITS search, Foreign Language Teaching (FLT) insights, real-life classrooms 2/ address real format education needs using NLP term. 3. **NLP appli in context of ITS**: ① **NLP for well-formed language**: **Need of users**: accurate explanation of grammar rules, vocab, structures, **NLP use**: analyse contexts/generate explanation, feedback, personnised assistance/interactive exercises and quizzes. ② **NLP for mal-formed**. **Needs**: correction feedback. Practice. **NLP used**: error detection, generate feedback. Interactive exercise by NLP enhanced ITS. 4. **Evaluation**: same as Text search in P.R.F.

9. Dialogue System. 1. **speech acts**: actions performed through language **common grounds**: info mutually shared by participants. **Turn taking**: roles of speaker & hearers. **Adjacency pairs**: sequence of two turns with expected structure. 作用 in DS: facilitate effective communication (recognize intentions, coherence in conversation, regulate conversation flow, maintain conversational structure). 2. **Grice Maxims**: norms guiding conversations ① **Quality**: say true things. ② **Quantity**: as much as info necessary, no more, no less ③ **Relevance**: say only rele. ④ **manner**: easily understood. (作用: norms guiding conversation, ensuring coherence and effectiveness, satisfying user experience.
3. **Task-Specific DS**: designed to help acomplish ≥ 1 specific concrete task