Ling/CSE 472: Introduction to Computational Linguistics

4/13
Evaluation & Error Analysis

Overview

- Evaluation in computational linguistics
- Error analysis: One nice example
- Term project specs
- Reading questions
- NLP/compling in the news

Why Evaluation?

- Good evaluation is essential to NLP research:
    - Verifies performance of process
    - Provides feedback on system changes
    - An essential part of the development process
    - Necessary for system comparisons
    - Provides information to potential users (and funders)

Ingredients

- Gold standard (“ground truth”)
- Evaluation metric: What you’ll count
- Baseline or baselines: What you’ll compare against
- Upper bound (optional)

Design considerations

- What system component is being evaluated? ex:
  - Parser
  - Language model
  - POS tagger
- What is the application? ex:
  - automated email response
  - travel dialogue system
  - document retrieval

Design considerations

- What are the evaluation criteria?
  - Accuracy
  - Coverage
  - Speed
  - Efficiency
  - Compatibility
  - Modifiability
  - Ease of use
  - Cost (in $ or carbon budget)
  - …

Design considerations

- What is the goal of the evaluation?
    - Validation: Does the system do what you meant it to do?
    - Regression testing: Do recent changes improve performance, and/or lose any coverage?
    - Intrinsic evaluation: How well does it perform the specific task?
    - Extrinsic evaluation: How does it impact overall system performance?
    - Hypothesis testing: Can X information be used to aid in Y task?

Design considerations

- What resources are available?
  - Annotated corpora (e.g., Treebanks, aligned corpora)
  - Specialized corpora from application domain
  - Dictionaries and lexicons (e.g., pronunciation dictionaries, WordNet)
  - Test suites
    - Systematic collections of acceptable and unacceptable examples of specific phenomena
    - Generally hand built for each system and evaluation
    - Efforts to create shared resources, e.g. TSNLP (English, French, German)
- Are there standard corpora or evaluation metrics for the task?

Data for evaluation

- Separate test data from training and development data
- Use standard data sets where possible, to facilitate replication of results and inter-system comparison
    - Data often the result of challenges or shared tasks sponsored by NIST or various workshops
    - Data often distributed through LDC or ELRA, or more recently Kaggle
- Where there is no standard, clearly define the data and make it available to others

Handling data: Machine learning paradigm

- Divide data into training, development and test sets:
  - Training: Original input to stochastic model
  - Development: “Pretest” for tuning parameters (to avoid over-fitting on training data)
  - Test: Held-out data to measure generalizability of the system
- Dev and test data are always annotated (“gold standard”)
- Training data may be annotated (supervised learning) or not

Handling data:
Knowledge engineering/rule-based paradigm

- “Training” data is examined by developer for rule development
- Training data is also used for regression testing
  - Does the current system analyze the same items as the previous one did?
  - Does the current system assign the same analyses as the previous one did?
- Test data is ideally unseen by both the system and the developer

Handling data:
Knowledge engineering/rule-based paradigm

- Dealing with out-of-vocabulary words:
    - Measure overall performance anyway
    - Select only test data with known vocabulary
    - Add lexical entries for unknown words and test remaining system
- Error analysis can be very informative

Evaluation metrics

- Quantifiable measures
- Human inspection may be best, but can be impractical
- Automated approximations are cheaper, and especially valuable during system development
- The best metrics are those aligned with the goals of the application
- Use standardized metrics where available
- If none are available, clearly define the metrics used and use more than one

Example Metric: Precision and Recall

- Originally developed (and named) for Information Retrieval as a metric for search effectiveness

- Extended to the evaluation of various NLP tasks, especially ones involving categorization/labeling

- Provides measures of how correct (precision) and how thorough (recall); these goals are usually in tension

Precision and Recall

- Precision:
  - Proportion of results of the system that were correct

$$P = \frac{\#\text{correct results}}{\#\text{results returned}}$$

- Recall:
  - Proportion of correct results that were returned by system

$$R = \frac{\#\text{correct results}}{\#\text{results in gold standard}}$$

F-measure (combination of P and R)

$$F = \frac{(\alpha + 1) \times P \times R}{\alpha P + R}$$

- Varying the constant α affects the weight of Precision vs. Recall; increasing α increases the weight of Recall in the measure
- If α =1, Precision and Recall are equally weighted:

$$F = \frac{2 \times P \times R}{P + R}$$

Precision and Recall: Questions

- Tasks
  - Part of speech dictionary construction
  - Morpheme boundary detection
  - Word sense disambiguation
- Questions
  - What would the gold standard data be?
  - What does precision mean here?
  - What does recall mean here?

Precision and Recall: Questions

- Why do we need to measure both precision and recall?

- Why would precision and recall be in competition?

Applications where precision is important

Total Results: 0

Applications where recall is important

Total Results: 0

Example Metric: BLEU score

- Automatic evaluation metric for machine translation (MT) (Papineni et al, ACL 2002)

- Measures similarity between system output and reference translations (gold standard)

- Measures lexical choice (unigrams), fluency (n-grams), and something like syntax (n-grams)

- Weighted average of the number of n-gram overlaps with reference translations: Weighted geometric mean of unigram, bigram, trigram and 4-gram scores

BLEU score

- Useful for comparing MT systems and tracking systems over time
- No meaningful units; for comparison, data sets must be the same
- One of several automatic MT evaluation metrics useful for development feedback
- Oft criticized
- Best MT evaluations use human raters (fluency, adequacy, edit distance)

Example metric: Parseval

- Automatic metric for evaluating parse accuracy when an annotated corpus is available

- Compares parser output to reference parses (gold standard)

- Evaluates component pieces of a parse

- Does not require an exact match: gives credit for partially correct parses

Parseval measures

- Labeled precision:

$$\frac{\text{\# of correct constituents in candidate parse}}{\text{total \# of constituents in candidate parse}}$$

- Labeled recall:

$$\frac{\text{\# of correct constituents in candidate parse}}{\text{total \# of constituents in gold standard parse}}$$

  - Constituents defined by starting point, ending point, and non-terminal symbol of spanning node

- Cross brackets: average number of constituents where the phrase boundaries of the gold standard and the candidate parse overlap

  - Example overlap: ((A B) C) v. (A (B C))

Issues with Parseval

- Parseval is the standard metric. However:
- Flawed measure:
    - Not very discriminating -- can do quite well while ignoring lexical content altogether
    - Sensitive to different styles of phrase structure (does particularly well on the flat structure of the Penn Treebank)
    - Too lenient sometimes, too harsh at others
    - Single errors may be counted multiple times
- Relevant only for CFGs (Phrase Structure Grammars)
- Most important question is: How well does it correlate with task improvement? Not clear.

Comparison

- Baseline: What you must beat
- Competing systems: What you want to beat
- Upper Bound (ceiling): What you aspire to
- Any difference must be statistically significant to count
- When comparing components, the rest of the system must be kept constant

LLMs and evaluation: discuss with a neighbor

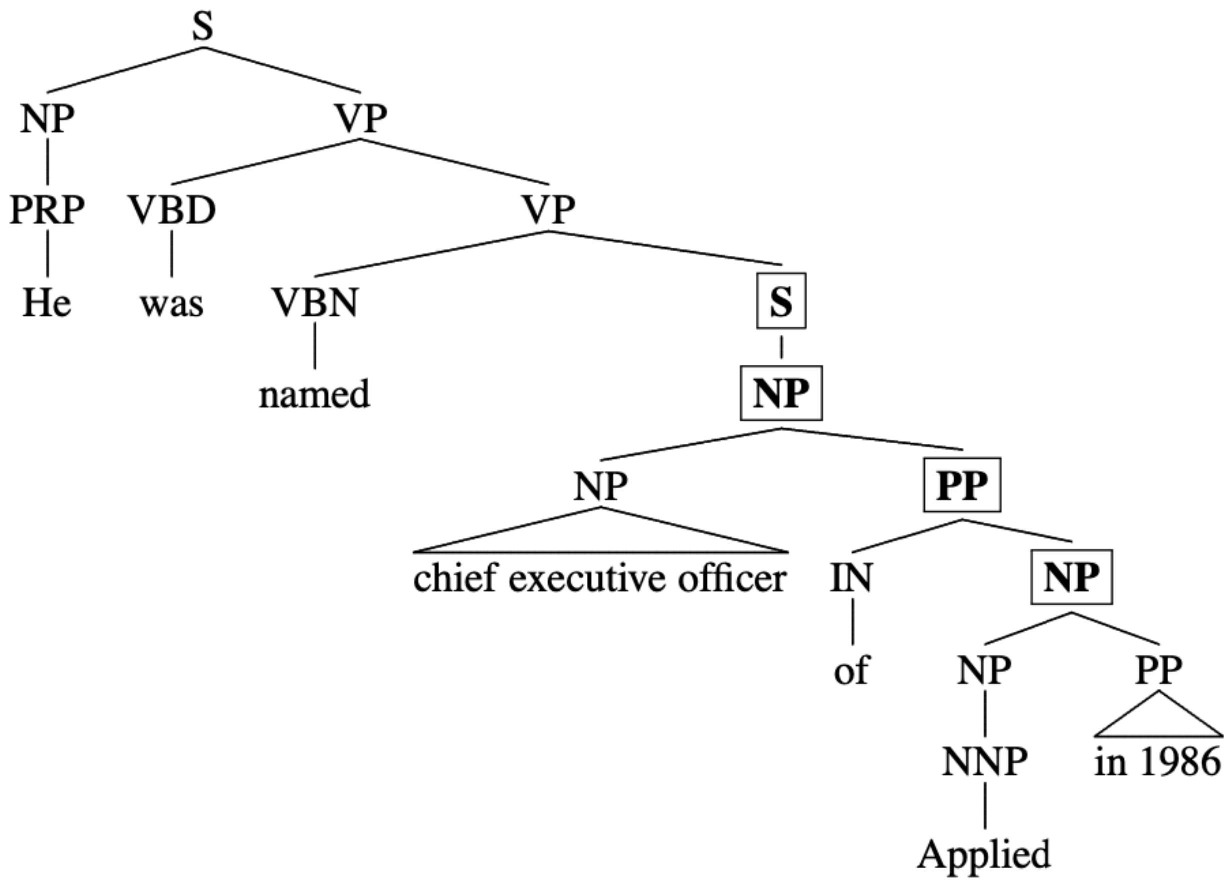
- How would you evaluate an LLM?
- How would you evaluate a chatbot?

Error analysis

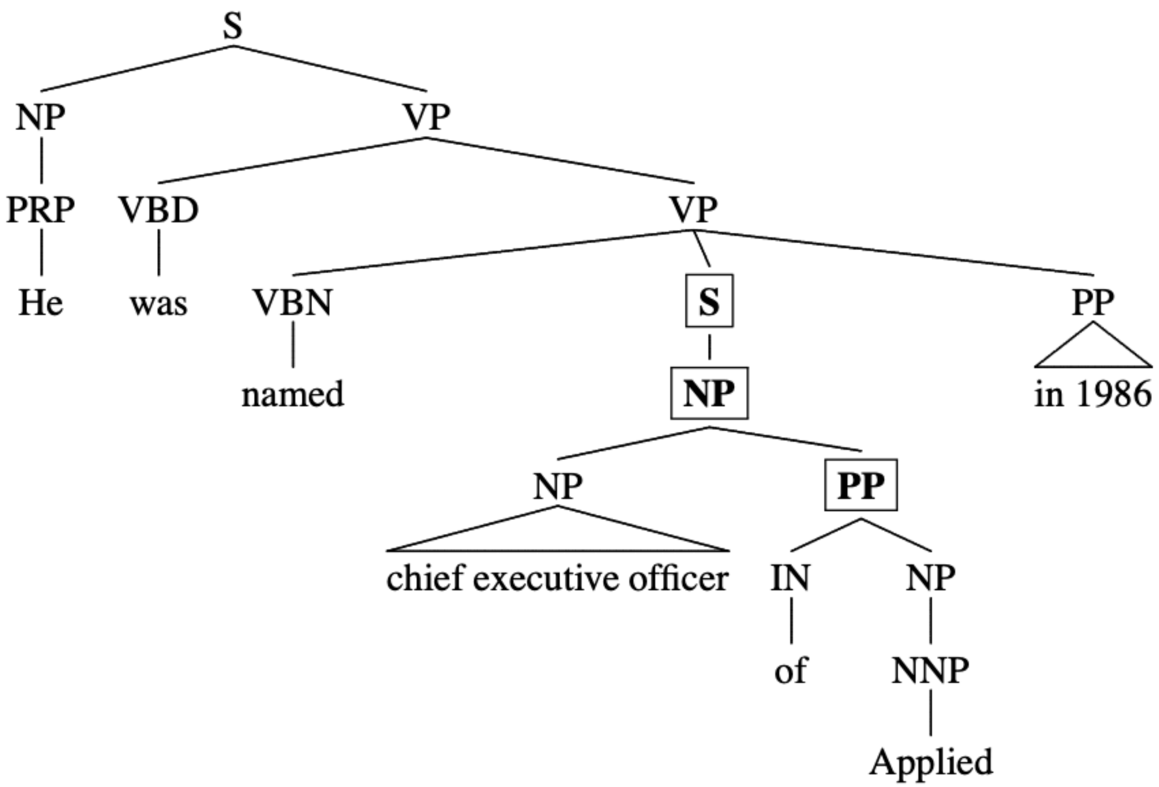
- What types of errors does the system make?
- What are the likely causes of each error type?
- How could the system be improved?
    - Which changes would have the most impact?
- How do the errors affect larger system performance?
- Note difference between error analysis and debugging

Kummerfeld et al 2012

- Target of evaluation: Constituent parsing of English
- Training data: Hand-labeled constituent trees (WSJ text, Penn Treebank)
- Output: Parse trees (CFG-style)
- Different parsers = different parse selection strategies
- Standard evaluation metric: Parseval



(a) Parser output



(b) Gold tree

Kummerfeld et al 2012

- Group individual errors at the constituent level into patterns representative of larger ‘mistakes’

- Look across those mistake types for better understanding of:

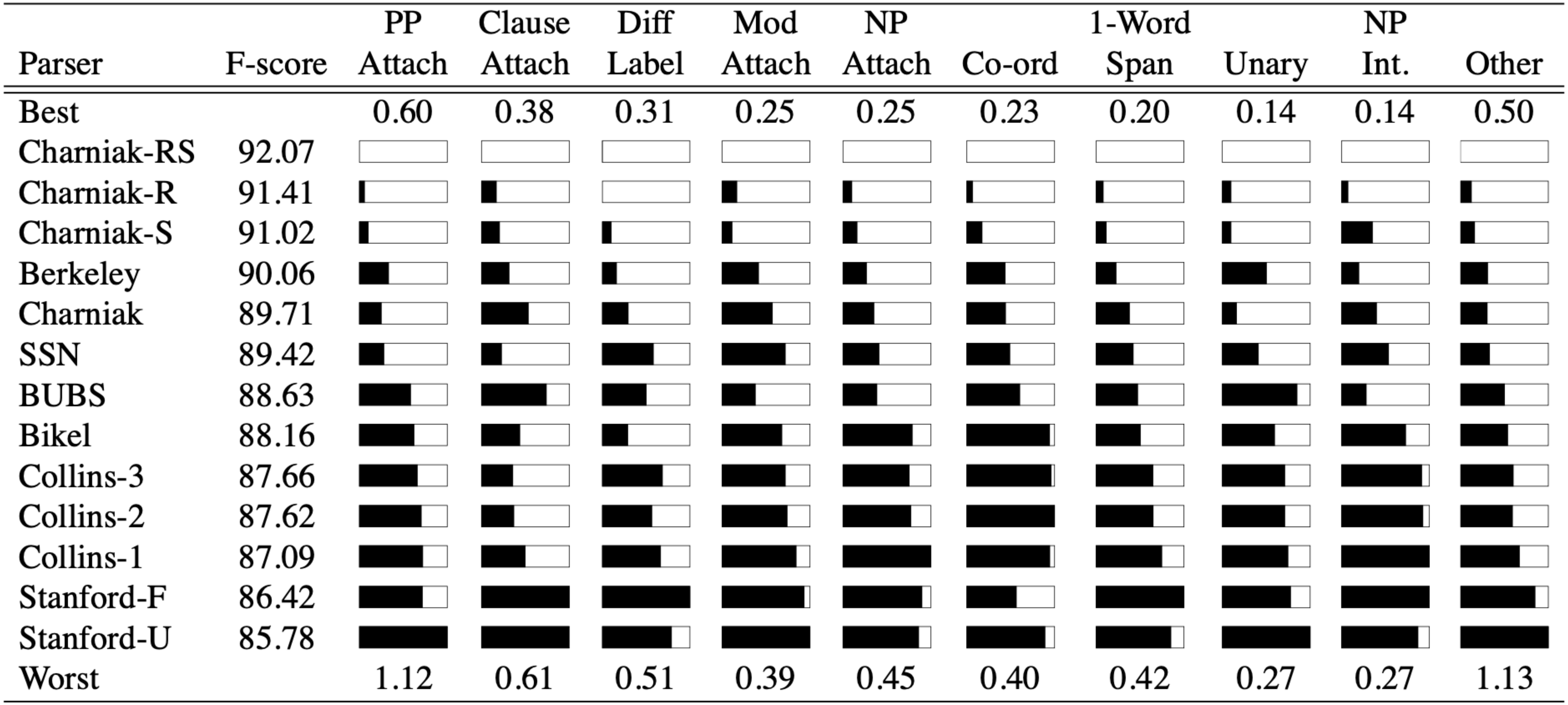
  - What is hard about constituent parsing of English

  - Strengths of different algorithms

  - Where the difficulties arise in cross-domain parsing

Error Type	Occurrences	Nodes Involved	Ratio
PP Attachment	846	1455	1.7
Single word phrase	490	490	1.0
Clause Attachment	385	913	2.4
Modifier Attachment	383	599	1.6
Different Label	377	754	2.0
Unary	347	349	1.0
NP Attachment	321	597	1.9
NP Internal Structure	299	352	1.2
Coordination	209	557	2.7
Unary Clause Label	185	200	1.1
VP Attachment	64	159	2.5
Parenthetical Attachment	31	74	2.4
Missing Parenthetical	12	17	1.4
Unclassified	655	734	1.1

Table 3: Breakdown of errors on section 23 for the Charniak parser with self-trained model and reranker. Errors are sorted by the number of times they occur. Ratio is the average number of node errors caused by each error we identify (i.e. Nodes Involved / Occurrences).



Parser	F-score	PP Attach	Clause Attach	Diff Label	Mod Attach	NP Attach	Co-ord	1-Word Span	Unary	NP Int.	Other
Best		0.60	0.38	0.31	0.25	0.25	0.23	0.20	0.14	0.14	0.50
Charniak-RS	92.07										
Charniak-R	91.41										
Charniak-S	91.02										
Berkeley	90.06										
Charniak	89.71										
SSN	89.42										
BUBS	88.63										
Bikel	88.16										
Collins-3	87.66										
Collins-2	87.62										
Collins-1	87.09										
Stanford-F	86.42										
Stanford-U	85.78										
Worst		1.12	0.61	0.51	0.39	0.45	0.40	0.42	0.27	0.27	1.13

Table 2: Average number of bracket errors per sentence due to the top ten error types. For instance, Stanford-U produces output that has, on average, 1.12 bracket errors per sentence that are due to PP attachment. The scale for each column is indicated by the Best and Worst values.

Kummerfeld et al 2012

- Group individual errors at the constituent level into patterns representative of larger ‘mistakes’

- Look across those mistake types for better understanding of:

  - What is hard about constituent parsing of English

  - Strengths of different algorithms

  - Where the difficulties arise in cross-domain parsing

Overview

- Evaluation in computational linguistics
- Error analysis: One nice example
- Term project specs
- Reading questions
- NLP/compling in the news

Term project specifications

- https://courses.washington.edu/ling472/final_project.html

Reading questions

- The paper evaluates parsers using metrics such as F-score, precision, and recall, but what are these metrics representative of? I didn't really understand what these words meant based on my understanding of linguistics.

Reading questions

- What are most algorithm developers using today - human or automated error analysis? I would assume that automated error analysis has probably gotten better since this paper was written, but I would still think that human analysis of the algorithm's outputs would almost be more valuable even if it's more expensive/time intensive since we know our languages best?

- Is there any framework put in place for evaluating evaluations? Especially automatic evaluations? How ‘human-involved’ are automatic evaluations?

Reading questions

- I would assume that that automatic evaluations are ultimately created by humans--do they produce different results than manual evaluations, and if so, how?

- Manual systems do have problems associated with it, but is there any way to transition from manual to automation with any mistakes since automation might make mistakes at the initial stages of the AI learning.

Reading questions

- As the book talked about so many different evaluations and each is perfomed in difference stage of the development, is there is gold standard process in the development process that which evaluation should be used? In addition, how ill-performed evaluation could make the algorithm go to the wrong route.

Reading questions

- It seems like asking researchers to include error metrics (or even general info on what went "wrong") could potentially result in inconsistencies; researchers may be hesitant (consciously or subconsciously) to include such information for a number of reasons as discussed in van Miltenburg paper. This speaks to the importance of consistent and uniform error evaluation methods which could potentially be introduced as a standard to include in relevant research. How common is the use of a system like Errudite for the purposes of uniform error analysis across different linguistic metrics? Are there other alternatives that have seen consistent use as a new "gold standard metric" across a wide amount of research, and if not, why not?

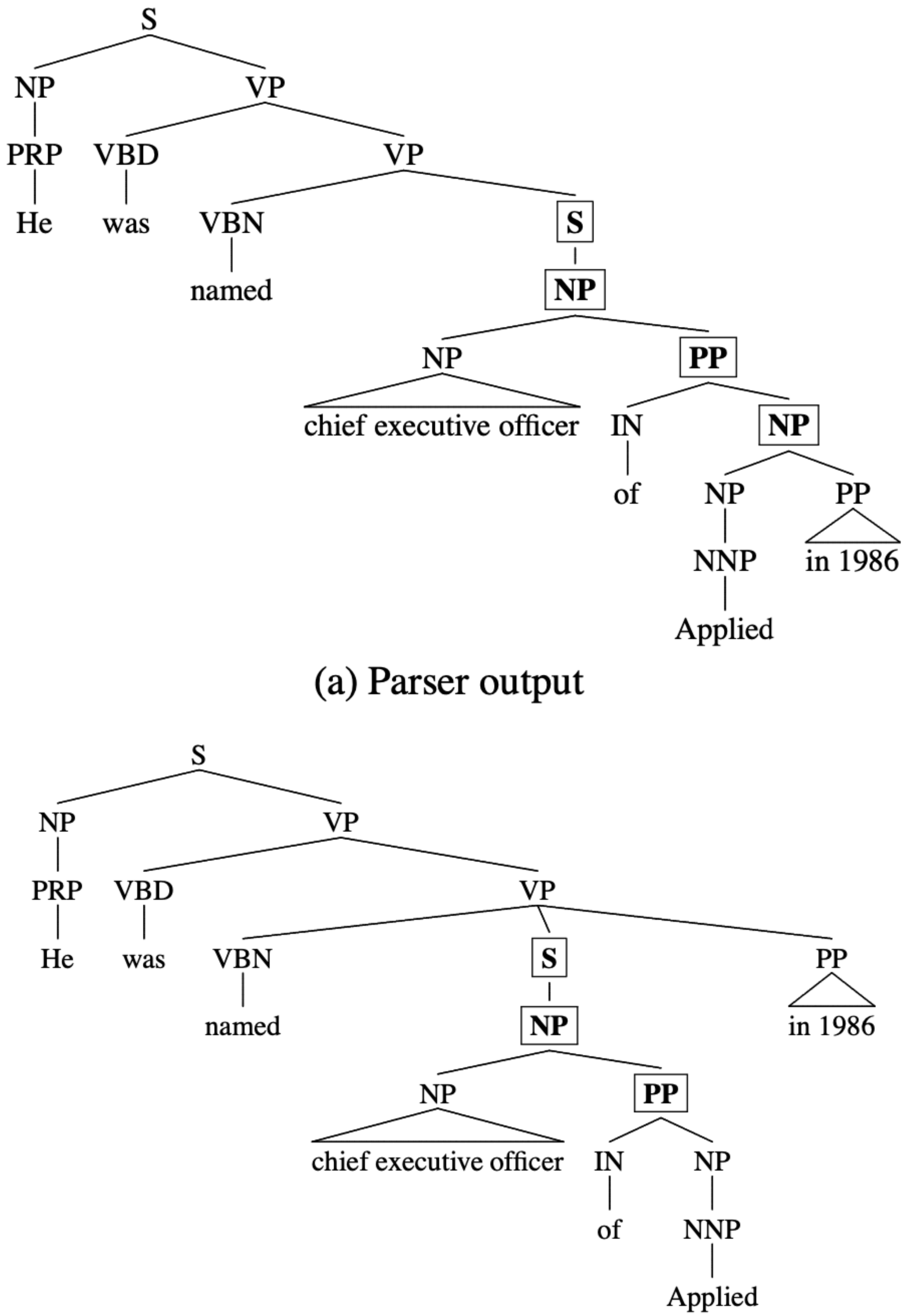
Reading questions

- What are the most common types of errors in parser output?

- What are coordination and attachment errors? Labeling errors made sense to me but I had a hard time understanding what it meant for errors to result from coordination and clause misattachments.

Reading questions

- This is from Figure 1 in Kummerfeld et al (Gold tree = correct parse). I'm confused about the lower S node - why does it only contain an NP without a VP, and is this common? I think the tree would be perfectly functional and possibly even make more sense with just an NP, not an S, representing "chief executive officer of Applied".



(a) Parser output

(b) Gold tree

Reading questions

- How or why are syntax trees like the one shown in Kummerfeld et al. used in NLP/computational linguistics? Are they used solely for error classification/ parsers? I'm also a bit confused about how that works exactly.

- Are other types of trees such as binary trees often used?

NLP/Compling in the news

- https://arstechnica.com/gadgets/2023/04/generative-ai-is-cool-but-lets-not-forget-its-human-and-environmental-costs/

- https://www.caidp.org/cases/openai/