Natural Language Processing for Low-Resource and Marginalized Language Varieties

Pillar III: Evaluation

Sina Ahmadi (sina.ahmadi@uzh.ch)

Department of Computational Linguistics

March 11, 2026

Adapted with thanks from Zdeněk Žabokrtský's "Evaluation measures in NLP" (NPFL124, Charles University)

Recap

NLP Framework

- Pillar I on Data: How to collect data?
- Pillar II on Learning: How to learn a model?
- **Pillar III on Evaluation: How do we know whether our systems actually work?**

Evaluation in NLP

What is evaluation for?

- The goal of NLP evaluation is to measure one or more qualities of an algorithm or system
- A useful side effect: defining proper evaluation criteria helps *specify the NLP problem precisely*
- Evaluation requires two things:
  - **Evaluation metrics** $\rightarrow$ what to measure
  - **Evaluation data** $\rightarrow$ what to measure on

Automatic vs. manual evaluation

Manual (human) evaluation:

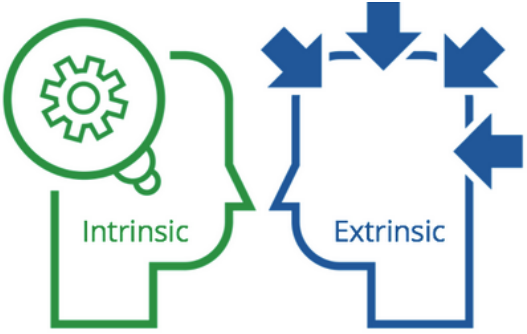
- Human judges assess system quality based on defined criteria
- Necessary when gold standards are hard to define (e.g., low inter-annotator agreement)

Automatic evaluation:

- Compare system output against a gold standard
- Cost of producing gold data can be high, but evaluation is then cheaply repeatable

Intrinsic vs. extrinsic evaluation

- Intrinsic evaluation:
  - Evaluates the NLP system in isolation
  - Example: measuring POS tagging accuracy
- Extrinsic evaluation:
  - Evaluates the system as a component in a larger pipeline
  - Example: does better POS tagging improve downstream MT quality?



The naive experiment loop

The simplest approach:

1. Train a model on training data
2. Evaluate on evaluation data
3. Improve the model
4. Go to 1

What's wrong with this?

The more iterations you run, the more the evaluation data effectively becomes part of your training signal

A better data division

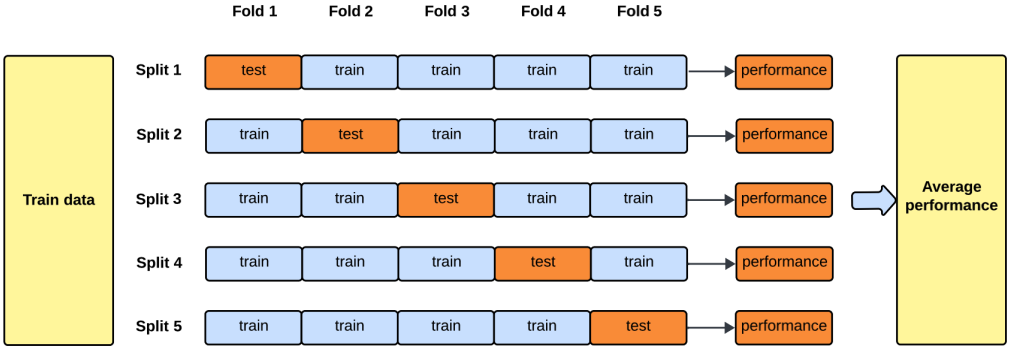
- Divide your data into **three** portions:
  - **Training data**: for model learning
  - **Development data** (devset / devtest): for tuning and iterative evaluation
  - **Test data** (etest): to be used *only once* for final evaluation
- Sometimes even more splits are needed (e.g., for hyperparameter selection)

When data is extremely scarce, even a three-way split may leave each portion too small to be reliable

K-fold cross-validation

Especially useful when data is very small:

1. Partition data into K roughly equal subsets (typically $K = 10$)
2. For each fold: train on $K - 1$ subsets, test on the remaining one
3. Average the K results



Core Metrics: accuracy

$$\text{accuracy} = \frac{\text{correctly predicted instances}}{\text{all instances}} \times 100\%$$

- "Correctly predicted" = identical to human decisions in annotated data
- Example: POS tagging:
  - An expert annotates POS tags in a corpus
  - Data is split: one part for training, one for evaluation
  - Accuracy = ratio of tokens with correctly predicted POS values
- Assumption: all errors are equally severe. Is that realistic?

When accuracy is not enough

Accuracy works well when:

- The number of task instances is known
- There is exactly one correct answer per instance
- The system gives exactly one answer per instance
- All errors are equally wrong
- But what if these assumptions don't hold? $\Rightarrow$ **precision** and **recall**

Precision and recall

- **Precision**: when the system gives a prediction, how often is it right?

$$\text{precision} = \frac{\text{correct answers given}}{\text{all answers given}} = \frac{TP}{TP + FP}$$

- **Recall**: what proportion of all correct answers does the system find?

$$\text{recall} = \frac{\text{correct answers given}}{\text{all possible correct answers}} = \frac{TP}{TP + FN}$$

F-measure

F-measure = weighted harmonic mean of precision (P) and recall (R):

$$F_\beta = \frac{(\beta^2 + 1) \cdot P \cdot R}{\beta^2 \cdot P + R}$$

Most commonly, equal weighting ($\beta = 1$):

$$F_1 = \frac{2 \cdot P \cdot R}{P + R}$$

- Balances the trade-off between precision and recall
- $\beta > 1$ favors recall; $\beta < 1$ favors precision

Exercise: Precision, Recall, F1

A sentiment classifier labels movie reviews as **positive** or **negative**.

Gold:	P	P	N	P	N	N	P	N	P	N
System:	P	N	N	P	P	N	P	N	N	N

- How many true positives (TP)? **TP = 3** (positions 1, 4, 7)
- How many false positives (FP)? **FP = 1** (position 5)
- How many false negatives (FN)? **FN = 2** (positions 2, 9)

$$\text{Precision} = \frac{\text{correct positive predictions}}{\text{all positive predictions}} = \frac{3}{4} = 75\%$$

$$\text{Recall} = \frac{\text{correct positive predictions}}{\text{all actual positives}} = \frac{3}{5} = 60\%$$

$$F_1 = \frac{2 \cdot P \cdot R}{P + R} = \frac{2 \cdot 0.75 \cdot 0.60}{0.75 + 0.60} = 66.7\%$$

When the system says "positive", it is right 75% of the time, but it only finds 60% of all actual positives.

$$\text{Accuracy} = \frac{\text{correctly predicted instances}}{\text{all instances}} = \frac{7}{10} = 70\%$$

Word Error Rate (WER)

A common metric for speech recognition:

$$\text{WER} = \frac{S + D + I}{S + D + C}$$

where:

- $S =$ substituted words
- $D =$ deleted words
- $I =$ inserted words
- $C =$ correct words

WER can be misleadingly high when transcription norms are unstable or when code-switching is present

BLEU (Bilingual Evaluation Understudy) (Papineni et al., 2002)

A common metric for machine translation:

$$\text{BLEU} = BP \cdot \exp\left(\frac{1}{4}\sum_{n=1}^{4}\log p_n\right)$$

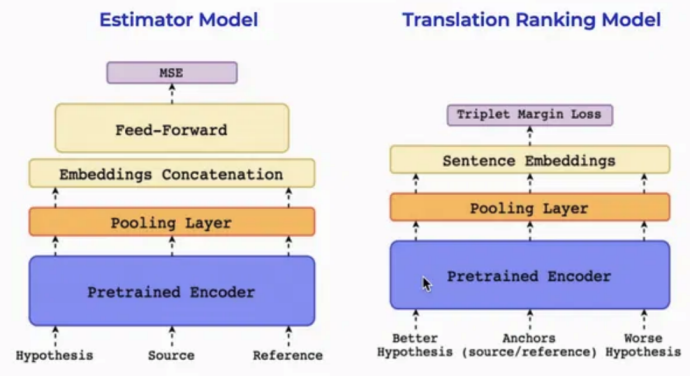
$$BP = \min\left(1, \exp\left(1 - \frac{r}{c}\right)\right)$$

- $p_n = n$-gram precision (with reference-clipped counts)
- $BP =$ brevity penalty (penalizes overly short translations)
- $r =$ reference length, $c =$ candidate length

Limitations assumes a single reference is representative; struggles with morphologically rich languages and free word order

Trainable Metrics: COMET (Rei et al., 2020)

- Unlike BLEU, which relies on surface-level n-gram overlap, **COMET** is a learned metric that uses a multilingual pre-trained encoder (e.g., XLM-R) to predict human quality judgments
- Trained on human evaluation data (e.g., direct assessments), it captures semantic similarity beyond lexical matching



Evaluation as trade-off

Complex evaluation metrics are often a compromise between competing criteria:

- **F-measure**: precision vs. recall
- **BLEU**: n-gram precision vs. brevity penalty
- **Manual MT evaluation**: fluency vs. adequacy
- **COMET**: semantic similarity vs. fidelity to human judgments

There is no single "correct" metric $\rightarrow$ the choice depends on the task, the language, and the application context

Interpreting Results

Is your result good or bad?

- You run an experiment, get a number. . . is it good?
- No easy answer, but you can interpret it relative to:
  - **Lower bound**: performance of a baseline (a simple or trivial system)
  - **Upper bound**: oracle experiment or inter-annotator agreement
- The range 0–100% does not mean every value in that interval is reasonable

Baselines

A sequence of increasingly complex reference systems:

- Random-choice baseline
- Most-frequent-value baseline
- GPT-4 in a zero-shot setup
- Intermediate solutions (e.g., unigram model for POS tagging: $\arg\max P(\text{tag} \mid \text{word})$)
- The previous state-of-the-art result

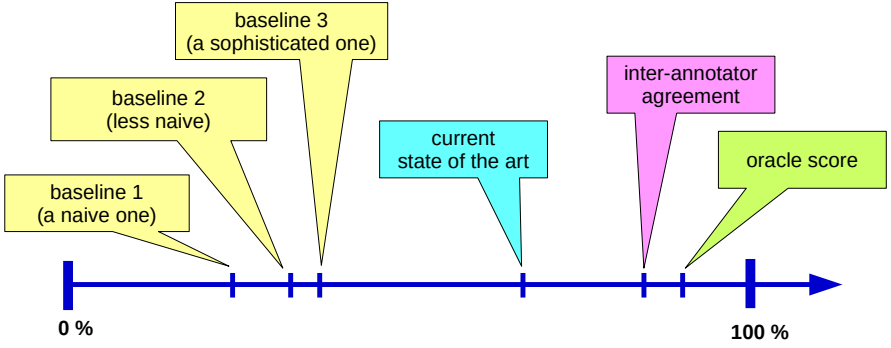
Oracle experiments

- Purpose: find the performance **upper bound** from the evaluated component's perspective
- An **oracle** makes the best possible decisions, *while respecting other limitations of the setup*
- *what's the best score a system could achieve if it always made the optimal choice, but was still subject to the other constraints of your pipeline?*
- Oracle performance $< 100\%$ reveals structural limitations in the pipeline

Example: oracle in POS tagging:

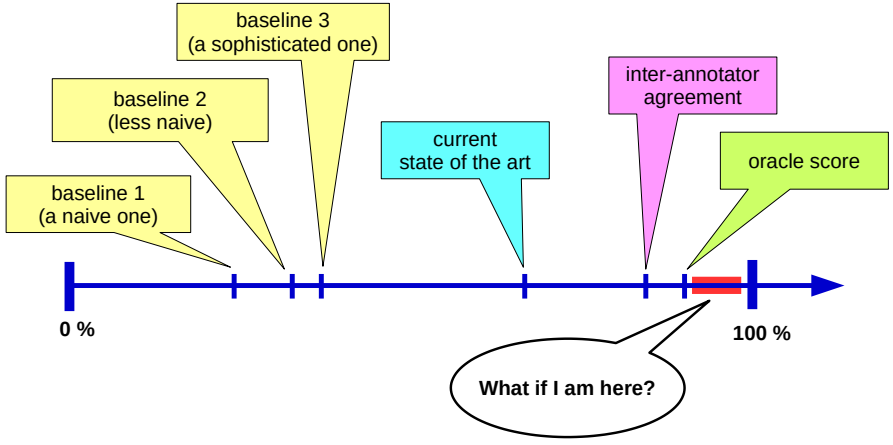
- Collect all possible POS tags per word from an annotated corpus
- On test data, select the correct tag whenever it appears in the candidate set

Evaluation in context



Always situate your result on a scale from baselines to oracle/IAA; a raw number alone is meaningless

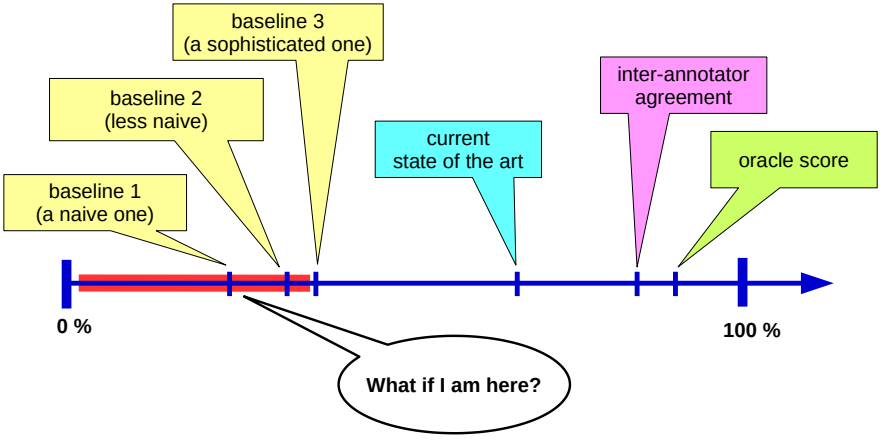
What if I'm here? Above the oracle



Outperforming an oracle is *impossible by definition*.

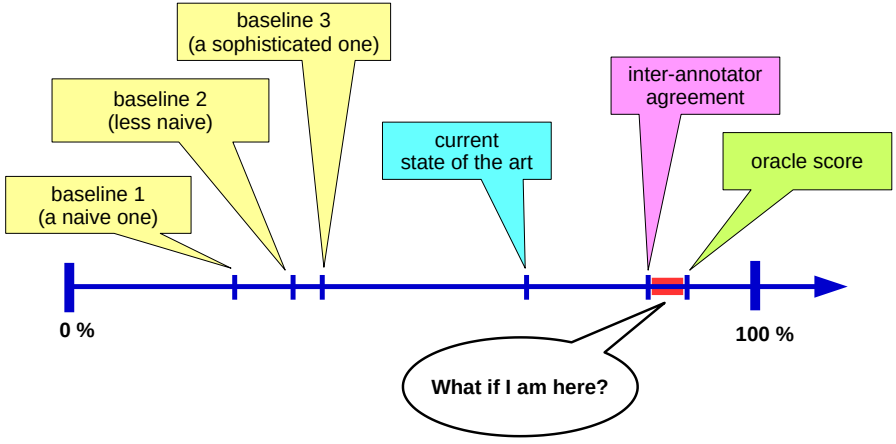
$\Rightarrow$ Search for a bug in your evaluation script.

What if I'm here? Below all baselines



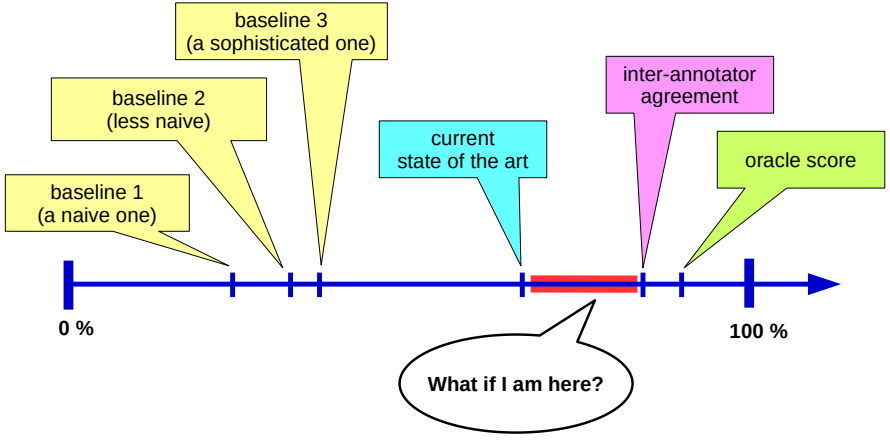
Something is likely wrong: model architecture, hyperparameters, underfitting/overfitting, or bugs. Apply standard ML diagnostics before publishing.

What if I'm here? Above IAA



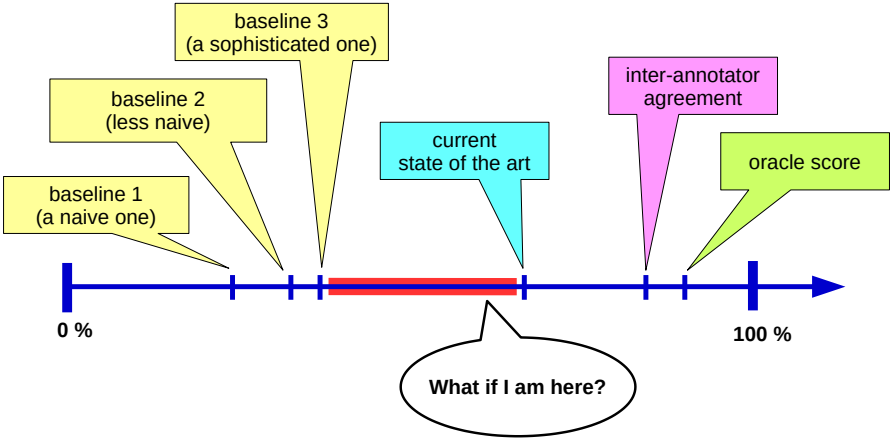
Possible but suspicious! In most NLP tasks, humans still outperform machines. Check for data leakage (e.g., mixing training and evaluation data).

What if I'm here? New state of the art!



Congratulations! But double-check: is the setup fair? Did evaluation data leak into training? Did you evaluate too many times on the test set?

What if I'm here? The most common outcome



You beat the baselines but not the current SOTA; this is the most common result. Your approach may still be valuable for its speed, robustness in low-resource settings, or licensing.

Annotation Quality and Agreement

Noise in annotations

"Ground truth" is produced by human annotators, who introduce inconsistencies:

- Annotators need time to internalize guidelines
- Guidelines evolve with annotation experience
- Different annotators make different decisions, even under the same instructions
- Errors from fatigue, speed, insufficient focus

  $\Rightarrow$ It is **essential** to quantify inter-annotator agreement (IAA)

Inter-annotator agreement (IAA)

- IAA measures how reliably humans perform a given annotation task
- Simplest form: accuracy of one annotator against the other
- But is IAA $= 0.8$ good enough? We need to account for **agreement by chance**

Example: two annotators, two classes (A, B)

- Annotator 1: 80% A, 20% B
- Annotator 2: 85% A, 15% B
- They agree on 80% of items
- Agreement by chance: $0.8 \times 0.85 + 0.2 \times 0.15 = 71\%$

Cohen's Kappa

$$\kappa = \frac{P_a - P_e}{1 - P_e}$$

- P_a = observed agreement (probability that annotators agree)
- P_e = expected agreement by chance
- Scale from -1 to $+1$ (negative kappa is rare but possible)

Conventional interpretation:

- 0.40–0.59: weak agreement
- 0.60–0.79: moderate agreement
- 0.80–0.90: strong agreement

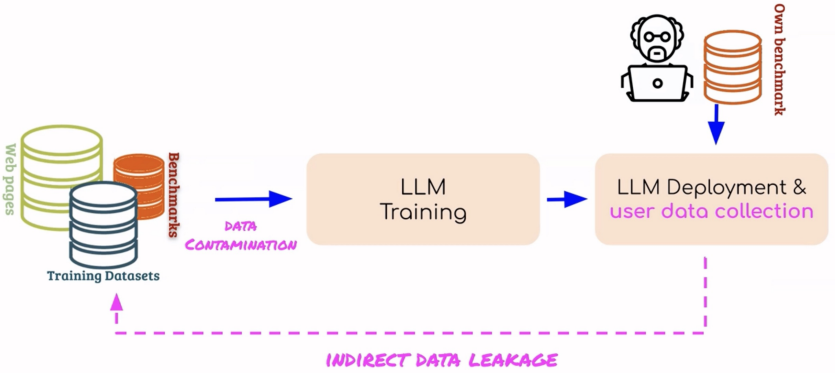
Our example: $\kappa = \frac{0.80-0.71}{1-0.71} = \frac{0.09}{0.29} = 0.31 \Rightarrow$ weak agreement, despite 80% raw agreement!

Significance

- When presenting results, you express two things:
  - The value itself
  - Your **certainty** about it
- **Statistical significance testing**: is the difference between two systems likely due to chance?
- Common tests: paired bootstrap resampling, approximate randomization
- Reporting confidence intervals helps readers judge whether improvements are meaningful

Data Contamination

- **Data contamination** occurs when evaluation benchmark data appears in a model's training data
- Balloccu et al. (2024) reviewed recent papers evaluating closed-source LLMs: around 42% had leaked data!



Takeaways

Takeaways

- Evaluation is perhaps the most important role of data in NLP
- Every metric rests on assumptions; even plain accuracy assumes all errors are equally severe
- Results vary across datasets, genres, and languages; always anchor your results in a bigger picture (baselines, SOTA, oracle, IAA)
- Don't just report numbers. *Interpret* them!

Reminders

Seminar Presentations (starting next week):

- Which paper do you present?

Final Project (50% of assessment):

- **Week 5** (next week): Problem Identification. Pick a language variety and a task.
- Week 8: Solution Proposal
- Week 14: Implementation & Presentation

References

Several slides in this lecture are adapted from Zdeněk Žabokrtský's lecture on "Evaluation Measures in NLP" (NPFL070, Charles University, 2022), licensed under CC BY-SA.

S. Balloccu, P. Schmidtová, M. Lango, and O. Dusek. Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source LLMs. In Y. Graham and M. Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 67–93, St. Julian's, Malta, Mar. 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.5. URL https://aclanthology.org/2024.eacl-long.5/.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL, 2002. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

R. Rei, C. Stewart, A. C. Farinha, and A. Lavie. COMET: A neural framework for MT evaluation. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.213. URL https://aclanthology.org/2020.emnlp-main.213/.