COSI-230B: Natural Language Annotation for Machine Learning

Lecture 10: Inter-Annotator Agreement III

Jin Zhao

Brandeis University

Spring 2026

Today's Agenda

1. Krippendorff's Alpha
2. Cohen's κ vs. Krippendorff's α
3. WAWA vs. Krippendorff's α
4. Agreement for sequence labeling (spans)

Why Do We Need Krippendorff's Alpha?

Fleiss' Kappa has limitations:

- Requires a *fixed* number of annotators per item
- Only handles categorical (nominal) labels
- Cannot handle missing data

Real-world problems:

- Item 1 labeled by 3 people, Item 2 by only 2 (someone was absent)
- Labels are ratings on a 1–5 scale ("4 vs. 5" is closer than "1 vs. 5")
- Some annotators skipped some items

Krippendorff's Alpha handles all of these.

Krippendorff's Alpha: Overview

Most flexible agreement measure

Advantages:

- Any number of annotators (can vary per item)
- Missing data allowed
- Works with different data types:
  - Nominal (categories: Pos, Neg, Neu)
  - Ordinal (rankings: 1st, 2nd, 3rd)
  - Interval (ratings: 1–5 scale)
  - Ratio (measurements: 0.5kg, 1.2kg) — rare in NLP

Guideline: $\alpha > 0.8$ reliable, $\alpha > 0.67$ acceptable

The Big Idea: Disagreement Instead of Agreement

Kappa measures agreement. Alpha measures disagreement.

Why? Because disagreement lets us ask "how wrong?"

- For categories: wrong is wrong (Pos $\neq$ Neg, that's it)
- For ratings: "4 vs. 5" is a small error, "1 vs. 5" is a big error
- Disagreement + distance functions = handles all data types

The formula:

$$\alpha = 1 - \frac{D_o}{D_e}$$

- D_o = observed disagreement (how much do annotators *actually* disagree?)
- D_e = expected disagreement (how much would they disagree *by chance*?)

Reading the Alpha Value

$$\alpha = 1 - \frac{D_o}{D_e}$$

Three scenarios:

1. **Perfect agreement** ($D_o = 0$): no disagreement at all

$$\alpha = 1 - \frac{0}{D_e} = 1$$

2. **Chance-level** ($D_o = D_e$): disagreement equals what you'd get randomly

$$\alpha = 1 - \frac{D_e}{D_e} = 0$$

3. **Systematic disagreement** ($D_o > D_e$): annotators disagree *more* than chance

$$\alpha < 0 \quad \text{(something is wrong!)}$$

Distance Functions: Measuring "How Wrong"

Alpha uses a distance function d^2 to measure disagreement between two labels:

- **Nominal** (categories): $d^2 = 0$ if same, 1 if different
  - Pos vs. Pos $= 0$, Pos vs. Neg $= 1$
- **Ordinal** (rankings): d^2 based on rank distance
- **Interval** (ratings): $d^2 = (c - k)^2$
  - Rating 4 vs. 5: $d^2 = 1$ (small)
  - Rating 1 vs. 5: $d^2 = 16$ (large)
- **Ratio** (measurements): $d^2 = \frac{(c-k)^2}{(c+k)^2}$

Today: We focus on **nominal** (simplest case)

Example Data

4 items, 3 annotators, labels Pos/Neg

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

Goal: Compute D_o, D_e, then α

Nominal distance: $d^2 = 0$ (same label), 1 (different label)

What Is Observed Disagreement (D_o)?

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

Intuition: Look at all pairs of labels *within each item*. What fraction of those pairs disagree?

$$D_o = \frac{\text{disagreeing pairs within items}}{\text{total pairs within items}}$$

Or equivalently:

$$D_o = 1 - \frac{\text{agreeing pairs within items}}{\text{total pairs within items}}$$

This is similar to Fleiss' P_i, but now we aggregate across all items into one single number instead of averaging per-item rates.

D_o: Counting Total Pairs (Denominator)

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

For each item with n_i annotators: ordered pairs $= n_i(n_i - 1)$

Item	Annotators (n_i)	Ordered pairs: $n_i(n_i - 1)$
1	3	$3 \times 2 = 6$
2	3	$3 \times 2 = 6$
3	3	$3 \times 2 = 6$
4	3	$3 \times 2 = 6$

$$\text{Total pairs} = \sum_i n_i(n_i - 1) = 6 + 6 + 6 + 6 = 24$$

Note: If items had different numbers of annotators, each row would have a different value — this is where Alpha handles missing data!

D_o: Counting Agreeing Pairs (Numerator)

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

For each item: count how many annotators chose each label (n_{ij}), then compute agreeing pairs $= \sum_j n_{ij}(n_{ij} - 1)$

Item	n_{Pos}	n_{Neg}	Agreeing pairs	Calculation
1	3	0	6	$3 \times 2 + 0 \times 0 = 6$
2	2	1	2	$2 \times 1 + 1 \times 0 = 2$
3	0	3	6	$0 \times 0 + 3 \times 2 = 6$
4	1	2	2	$1 \times 0 + 2 \times 1 = 2$

$$\text{Total agreeing pairs} = \sum_i \sum_j n_{ij}(n_{ij} - 1) = 6 + 2 + 6 + 2 = 16$$

D_o: Putting It Together

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

$$D_o = 1 - \frac{\text{agreeing pairs}}{\text{total pairs}} = 1 - \frac{\sum_i \sum_j n_{ij}(n_{ij} - 1)}{\sum_i n_i(n_i - 1)}$$

Plugging in:

$$D_o = 1 - \frac{16}{24} = 1 - \frac{2}{3} = \frac{1}{3} \approx 0.333$$

Meaning: About 33% of within-item annotator pairs disagree.

But is 33% a lot or a little?

We need to compare it to what we'd expect by chance $\rightarrow D_e$

What Is Expected Disagreement (D_e)?

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

Intuition: Pool *all* labels together, ignoring which item they belong to. If we randomly drew pairs from this pool, how often would they disagree?

Step 1: Pool all labels

- Item 1: Pos, Pos, Pos
- Item 2: Pos, Pos, Neg
- Item 3: Neg, Neg, Neg
- Item 4: Pos, Neg, Neg

Total labels: $N = 4 \times 3 = 12$

Step 2: Count each label

- $N_{\text{Pos}} = 3 + 2 + 0 + 1 = 6$ labels are Pos
- $N_{\text{Neg}} = 0 + 1 + 3 + 2 = 6$ labels are Neg

Computing D_e

If we pick two labels at random from the pool, what is the chance they match?

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

Agreeing pairs in the pool:

$$\sum_j N_j(N_j - 1) = N_{\text{Pos}}(N_{\text{Pos}} - 1) + N_{\text{Neg}}(N_{\text{Neg}} - 1) = 6 \times 5 + 6 \times 5 = 30 + 30 = 60$$

Total ordered pairs in the pool:

$$N(N-1) = 12 \times 11 = 132$$

Expected disagreement:

$$D_e = 1 - \frac{\sum_j N_j(N_j - 1)}{N(N-1)} = 1 - \frac{60}{132} = 1 - \frac{5}{11} = \frac{6}{11} \approx 0.545$$

Meaning: If labels were assigned randomly (keeping the same overall proportions), about 55% of pairs would disagree.

Computing Alpha

Formula:

$$\alpha = 1 - \frac{D_o}{D_e}$$

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg

$$\begin{aligned} D_o \text{ (observed disagreement)} &= \tfrac{1}{3} \approx 0.333 \\ D_e \text{ (expected disagreement)} &= \tfrac{6}{11} \approx 0.545 \end{aligned}$$

Computation:

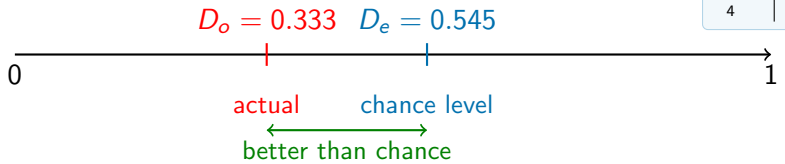
$$\alpha = 1 - \frac{1/3}{6/11} = 1 - \frac{1}{3} \times \frac{11}{6} = 1 - \frac{11}{18} = \frac{7}{18} = 0.389$$

Interpretation: $\alpha = 0.389$ — below the 0.67 threshold. Agreement is not reliable enough for this task.

Use Python: `pip install krippendorff`, then `krippendorff.alpha()`

Comparing D_o and D_e: Why It Works

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Pos	Neg
3	Neg	Neg	Neg
4	Pos	Neg	Neg



- Annotators disagree 33% of the time (actual)
- Random labeling would produce 55% disagreement (chance)
- So annotators are doing *better* than chance, but not by a lot
- $\alpha = 0.389$ captures this: some signal, but not reliable

Your Turn: Krippendorff's Alpha (Nominal)

5 items, 3 annotators, labels Pos/Neg

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Neg	Pos
3	Neg	Neg	Neg
4	Pos	Neg	Neg
5	Pos	Pos	Neg

Compute: D_o, D_e, and α

Formulas (Hint)

$$D_o = 1 - \frac{\sum_i \sum_j n_{ij}(n_{ij}-1)}{\sum_i n_i(n_i-1)}$$

$$D_e = 1 - \frac{\sum_j N_j(N_j-1)}{N(N-1)}$$

$$\alpha = 1 - \frac{D_o}{D_e}$$

Solution: Step 1 — Observed Disagreement (D_o)

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Neg	Pos
3	Neg	Neg	Neg
4	Pos	Neg	Neg
5	Pos	Pos	Neg

Label counts and agreeing pairs per item:

- Item 1: $n_{\text{Pos}}=3, n_{\text{Neg}}=0 \Rightarrow 3\times 2+0=6$
- Item 2: $n_{\text{Pos}}=2, n_{\text{Neg}}=1 \Rightarrow 2\times 1+1\times 0=2$
- Item 3: $n_{\text{Pos}}=0, n_{\text{Neg}}=3 \Rightarrow 0+3\times 2=6$
- Item 4: $n_{\text{Pos}}=1, n_{\text{Neg}}=2 \Rightarrow 1\times 0+2\times 1=2$
- Item 5: $n_{\text{Pos}}=2, n_{\text{Neg}}=1 \Rightarrow 2\times 1+1\times 0=2$

Agreeing pairs $= 6+2+6+2+2=18$

Total pairs: each item has $r=3$ annotators $\Rightarrow r(r-1)=3\times 2=6$ ordered pairs per item, $\times$ 5 items $= 5\times 6=30$

$$D_o = 1-\frac{18}{30}=1-\frac{3}{5}=\frac{2}{5}=0.40$$

Solution: Step 2 — Expected Disagreement (D_e)

Item	Jin	Maya	Sam
1	Pos	Pos	Pos
2	Pos	Neg	Pos
3	Neg	Neg	Neg
4	Pos	Neg	Neg
5	Pos	Pos	Neg

Pool all labels: $N = 5 \times 3 = 15$

- $N_{\text{Pos}} = 3 + 2 + 0 + 1 + 2 = 8$
- $N_{\text{Neg}} = 0 + 1 + 3 + 2 + 1 = 7$

Agreeing pairs in pool:

$$\sum_j N_j(N_j - 1) = 8 \times 7 + 7 \times 6 = 56 + 42 = 98$$

Total pairs in pool: $N(N-1) = 15 \times 14 = 210$

$$D_e = 1 - \frac{98}{210} = 1 - \frac{7}{15} = \frac{8}{15} \approx 0.533$$

Solution: Step 3 — Alpha (α)

Formula:

$$\alpha = 1 - \frac{D_o}{D_e}$$

$$\begin{aligned} D_o \text{ (observed disagreement)} &= \tfrac{2}{5} = 0.40 \\ D_e \text{ (expected disagreement)} &= \tfrac{8}{15} \approx 0.533 \end{aligned}$$

Computation:

$$\alpha = 1 - \frac{2/5}{8/15} = 1 - \frac{2}{5} \times \frac{15}{8} = 1 - \frac{30}{40} = 1 - \frac{3}{4} = 0.25$$

Interpretation: $\alpha = 0.25$ — well below the 0.67 threshold. The annotators agree only slightly better than chance.

Cohen's κ vs. α: A Revealing Example

10 items, 2 raters, labels Pos/Neg

	Maya = Pos	Maya = Neg	Row total
Jin = Pos	2	6	8
Jin = Neg	0	2	2
Col total	2	8	10

They agree on $2 + 2 = 4$ items out of 10:

$$P_o = 0.4$$

Key observation: Jin says Pos 80% of the time, but Maya says Pos only 20%. The raters have **very different biases** — how do κ and α handle this?

Cohen's κ: Uses Each Rater's Own Marginals

Rater marginals:

- Jin: $P(\text{Pos}) = 0.8, \quad P(\text{Neg}) = 0.2$
- Maya: $P(\text{Pos}) = 0.2, \quad P(\text{Neg}) = 0.8$

Expected agreement (Cohen):

$$\begin{aligned} P_e &= P(\text{both say Pos by chance}) + P(\text{both say Neg by chance}) \\ &= \underbrace{P_{\text{Jin}}(\text{Pos})}_{0.8} \cdot \underbrace{P_{\text{Maya}}(\text{Pos})}_{0.2} + \underbrace{P_{\text{Jin}}(\text{Neg})}_{0.2} \cdot \underbrace{P_{\text{Maya}}(\text{Neg})}_{0.8} \\ &= 0.16 + 0.16 = 0.32 \end{aligned}$$

Kappa:

$$\kappa = \frac{0.4 - 0.32}{1 - 0.32} = \frac{0.08}{0.68} \approx 0.118$$

Cohen's $\kappa \approx 0.12$ (slightly above chance)

Cohen's chance model: "each rater independently follows *their own* label distribution"

α: Uses Pooled Marginals

Pool all labels across both raters (20 total judgments):

- Pos count $= 8 + 2 = 10 \quad \Rightarrow \quad p(\text{Pos}) = 0.5$
- Neg count $= 2 + 8 = 10 \quad \Rightarrow \quad p(\text{Neg}) = 0.5$

Expected agreement under α's chance model:

$$P_e^{(\text{pooled})} = 0.5^2 + 0.5^2 = 0.5$$

Alpha (in kappa-like form):

$$\alpha = \frac{P_o - P_e^{(\text{pooled})}}{1 - P_e^{(\text{pooled})}} = \frac{0.4 - 0.5}{1 - 0.5} = \frac{-0.1}{0.5} = -0.2$$

$\alpha = -0.20$ (worse than chance!)

α's chance model: "labels are drawn from a *single pooled* distribution"

Why They Disagree: The Fundamental Difference

Same data, very different results:

	Cohen's κ	Krippendorff's α
Value	$+0.12$	-0.20
Verdict	Slightly above chance	Worse than chance

The root cause: different chance models

- **Cohen's κ:** chance = each rater follows *their own* distribution
  - Accounts for individual rater biases
- **α:** chance = all labels drawn from *one pooled* distribution
  - Treats raters as interchangeable

When raters have different biases (one says Pos a lot, the other says Neg a lot), the pooled distribution averages them out — making expected agreement *higher* — so α penalizes the observed disagreement more harshly.

WAWA vs. Krippendorff's α: The Short Version

WAWA	**Krippendorff's α**
Measure **worker reliability** against a consensus	Measure **overall reliability** of a coding scheme

Those are **different goals.**

What Krippendorff's α Is Designed For

Krippendorff's α is a **global reliability coefficient**. It:

- Works with any number of raters
- Handles missing data
- Works for nominal / ordinal / interval scales
- Corrects for chance agreement

Its purpose is **psychometric**:

"Is this annotation scheme reliable?"

It assumes raters are *interchangeable* and *symmetric*.

It does **not** tell you which rater is good or bad.

What WAWA Is Designed For

WAWA (Worker Agreement With Aggregate):

1. Build an aggregate label (e.g., majority vote)
2. Measure each worker's agreement with that aggregate
3. Average across workers

Its purpose is **operational**:

"Are my workers behaving consistently with the consensus?"

It treats the aggregate as a **proxy for ground truth**.

That makes it very practical in MTurk pipelines.

The Key Conceptual Difference

Krippendorff's α asks:

"Is there reliable signal beyond chance in this dataset?"

WAWA asks:

"How close are workers to the consensus label?"

Those are **not** the same question.

Why α Is Often Not Ideal for MTurk Pipelines

(1) You don't actually care about chance correction

- Workers are not randomly labeling — you already filter low-quality workers
- Chance correction can make agreement look artificially low (especially with skewed labels)
- α can look "bad" even when majority vote works perfectly fine

(2) You want worker-level diagnostics

- α gives *one number* for the whole dataset
- WAWA gives per-worker agreement — a natural way to filter bad annotators
- α cannot identify bad workers

(3) Majority vote is your actual output

- In many NLP pipelines, you collapse annotations into majority vote
- WAWA directly evaluates alignment with what you're going to use
- α evaluates the raw disagreement structure — which may not matter once you aggregate

When to Use α vs. WAWA

Use Krippendorff's α when...

- Publishing a dataset paper
- You need a recognized reliability coefficient
- You want something reviewer-proof
- You care about chance correction
- You need scale flexibility (ordinal, interval, etc.)

Use WAWA when...

- Managing crowd quality
- You want to drop low-performing workers
- Building a dataset operationally
- You care about consensus stability

The blunt truth

α is a **measurement-theory** statistic. WAWA is a **crowdsourcing quality-control** statistic.

Running MTurk and keeping annotators aligned? $\rightarrow$ **WAWA**
Arguing your coding scheme is reliable in a methods section? $\rightarrow \alpha$

Agreement for Sequence Labeling

NER and span annotation require special measures

Two approaches:

1. **Token-level:** Treat each token as a classification
   - Use standard Kappa on BIO labels
   - Doesn't capture span-level structure
2. **Entity-level:** Compare extracted spans
   - Precision/Recall/F1 between annotators
   - Exact match vs. partial match

Token-Level Agreement: Example

Sentence: "Barack Obama visited New York City last Friday."

Each token gets a BIO label:

Token	Barack	Obama	visited	New	York	City	last	Friday
Jin	B-PER	I-PER	O	B-LOC	I-LOC	I-LOC	O	B-DATE
Maya	B-PER	I-PER	O	B-LOC	I-LOC	O	B-DATE	I-DATE

Token-level agreement: treat each column as an independent classification

- 8 tokens total; Jin and Maya agree on 5 (Barack, Obama, visited, New, York)
- Raw agreement $= 5/8 = 0.625$
- Then compute Cohen's κ on the full BIO label confusion matrix

Problem: "City" and "last" are single-token errors, but they break *two entire entities* (LOC and DATE) — token-level metrics miss this.

Token-Level κ: Step 1 — Confusion Matrix

Aligning Jin's and Maya's BIO labels token by token:

Token	Jin	Maya	Agree?
Barack	B-PER	B-PER	✓
Obama	I-PER	I-PER	✓
visited	O	O	✓
New	B-LOC	B-LOC	✓
York	I-LOC	I-LOC	✓
City	I-LOC	O	×
last	O	B-DATE	×
Friday	B-DATE	I-DATE	×

Observed agreement:

$$A_o = \frac{5}{8} = 0.625$$

Token-Level κ: Step 2 — Expected Agreement

Count each label's frequency (out of 8 tokens each):

Label	Jin	p_{Jin}	Maya	p_{Maya}
B-PER	1	1/8	1	1/8
I-PER	1	1/8	1	1/8
O	2	2/8	1	1/8
B-LOC	1	1/8	1	1/8
I-LOC	2	2/8	1	1/8
B-DATE	1	1/8	1	1/8
I-DATE	0	0/8	2	2/8

Expected agreement (Cohen's): $A_e = \sum_j p_{\text{Jin}}(j) \times p_{\text{Maya}}(j)$

$$A_e = \tfrac{1}{8}\cdot\tfrac{1}{8} + \tfrac{1}{8}\cdot\tfrac{1}{8} + \tfrac{2}{8}\cdot\tfrac{1}{8} + \tfrac{1}{8}\cdot\tfrac{1}{8} + \tfrac{2}{8}\cdot\tfrac{1}{8} + \tfrac{1}{8}\cdot\tfrac{1}{8} + \tfrac{0}{8}\cdot\tfrac{2}{8} = \tfrac{1+1+2+1+2+1+0}{64} = \tfrac{8}{64} = 0.125$$

Token-Level κ: Step 3 — Kappa

Formula:

$$\kappa = \frac{A_o - A_e}{1 - A_e}$$

$$A_o \text{ (observed agreement)} = 0.625$$
$$A_e \text{ (expected agreement)} = 0.125$$

Computation:

$$\kappa = \frac{0.625 - 0.125}{1 - 0.125} = \frac{0.500}{0.875} = 0.571$$

Interpretation: $\kappa = 0.571$ — moderate agreement at the token level

But remember: this hides the fact that 2 out of 3 entities have boundary disagreements. Entity-level metrics tell a different story...

Entity-Level Agreement

Comparing annotator spans:

Exact match:

- Spans must have identical boundaries AND type
- Strict but clear

Partial match (relaxed):

- Allow some boundary variation
- More forgiving of minor differences

Metrics:

$$P = \frac{|A \cap B|}{|A|}, \quad R = \frac{|A \cap B|}{|B|}, \quad F_1 = \frac{2PR}{P + R}$$

Where A = Jin's spans, B = Maya's spans

Entity-Level Agreement: Example Sentence

Sentence: "Barack Obama visited New York City last Friday."

Jin found 3 entities:

1. [Barack Obama] – PER
2. [New York City] – LOC
3. [Friday] – DATE

Maya found 3 entities:

1. [Barack Obama] – PER
2. [New York] – LOC
3. [last Friday] – DATE

Question: How well do they agree?

Entity-Level Agreement: Exact Match

Exact match: spans must have identical boundaries AND type

Entity	Jin	Maya	Exact match?
PER	[Barack Obama]	[Barack Obama]	✓
LOC	[New York City]	[New York]	× (boundary differs)
DATE	[Friday]	[last Friday]	× (boundary differs)

Sets:

- A = Jin's spans: $|A| = 3$
- B = Maya's spans: $|B| = 3$
- $A \cap B$ (exact matches): $|A \cap B| = 1$ (only "Barack Obama / PER")

Compute:

$$P = \frac{|A \cap B|}{|A|} = \frac{1}{3} = 0.333 \qquad R = \frac{|A \cap B|}{|B|} = \frac{1}{3} = 0.333$$

$$F_1 = \frac{2 \times 0.333 \times 0.333}{0.333 + 0.333} = \frac{0.222}{0.666} = 0.333$$

Why Exact Match Can Be Too Strict

Both Jin and Maya found the same 3 entities!

But exact match gives $F_1 = 0.333$ because:

- "New York City" vs. "New York" — same entity, slightly different boundary
- "Friday" vs. "last Friday" — same entity, slightly different boundary

Partial match: give credit for overlapping spans with the correct type

Entity	Jin	Maya	Partial match?
PER	[Barack Obama]	[Barack Obama]	✓ (exact)
LOC	[New York City]	[New York]	✓ (overlap)
DATE	[Friday]	[last Friday]	✓ (overlap)

With partial match: $|A \cap B| = 3$

$$P = \frac{3}{3} = 1.0 \qquad R = \frac{3}{3} = 1.0 \qquad F_1 = 1.0$$

Takeaway: Always report which matching criterion you used!

Partial Match: How It Works

Idea: Instead of all-or-nothing, score each match by how much the spans overlap.

Token overlap score for a matched pair of spans:

$$\text{score} = \frac{|\text{tokens in common}|}{|\text{tokens in union}|}$$

Our example at the token level:

Type	Jin's tokens	Maya's tokens	Overlap	Score
PER	{Barack, Obama}	{Barack, Obama}	2 / 2	1.00
LOC	{New, York, City}	{New, York}	2 / 3	0.67
DATE	{Friday}	{last, Friday}	1 / 2	0.50

Rule: Two spans can match only if they have the **same type** and share **at least one token**.

Partial Match: Computing P, R, F1

Sum up the overlap scores instead of counting 0/1 matches:

Total overlap score $= 1.00 + 0.67 + 0.50 = 2.17, \quad |A| = 3, \quad |B| = 3$

Partial-match precision and recall:

$$P = \frac{\text{total overlap score}}{|A|} = \frac{2.17}{3} = 0.72 \qquad R = \frac{\text{total overlap score}}{|B|} = \frac{2.17}{3} = 0.72$$

F1:

$$F_1 = \frac{2 \times 0.72 \times 0.72}{0.72 + 0.72} = 0.72$$

Compare: Exact $F_1 = 0.333$ vs. Partial $F_1 = 0.72$

Partial match better reflects that annotators mostly agree on the entities.

Key Takeaways

1. **Krippendorff's Alpha** is most flexible (any data type, missing data)
2. **Cohen's κ vs. α:** different chance models give different results
3. **WAWA vs. α:** different goals — worker QC vs. scheme reliability
4. **Choose α** for publishing reliability; **choose WAWA** for crowd QC
5. **Span agreement** needs both token-level and entity-level measures
6. **Exact vs. partial match:** always report which criterion you used

Questions?

Office Hours: Wednesdays 1-3pm, Volen 109

jinzhao@brandeis.edu