Panel: Context-Dependent Evaluation of Tools for NL RE Tasks: Recall vs. Precision, and Beyond

Daniel Berry, Jane Cleland-Huang, Alessio Ferrari, Walid Maalej, John Mylopoulos, Didar Zowghi

Vocabulary

CBS = Computer-Based System

SE = Software Engineering
RE = Requirements Engineering
RS = Requirements Specification

NL = Natural Language
NLP = Natural Language Processing
IR = Information Retrieval

HD = High Dependability

HT = Hairy Task

NLP for RE?

After Kevin Ryan observed in 1993 that NLP was not likely to ever be powerful enough to do RE, …

RE researchers began to apply NLP to build tools for a variety of *specific* RE tasks involving NL RSs

NLP for RE!

Since then, NLP has been applied to

- **abstraction finding,**
- **requirements tracing,**
- **multiple RS consolidation,**
- **requirement classification,**
- **app review analysis,**
- **model synthesis,**
- **RS ambiguity finding, and its generalization,**
- **RS defect finding**

These and others are collectively NL RE tasks.

Task Vocabulary

A *task* is an instance of one of these or other NL RE tasks.

A task *T* is applied to a collection of documents *D* relevant to one RE effort for the development of a CBS.

A *correct answer* is an instance of what *T* is looking for.

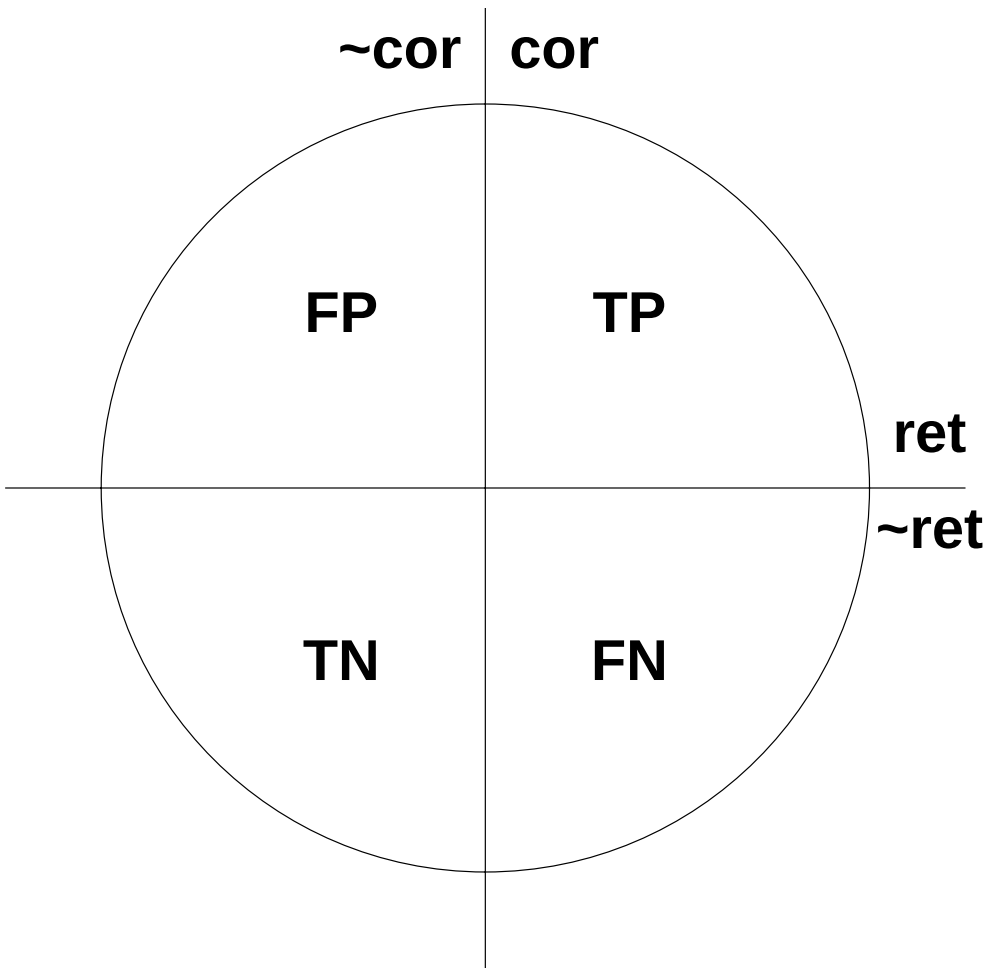
Task Vocabulary, Cont'd

A correct answer is somehow derived from *D*.

A *tool* for *T* returns to its users *answers* that it believes to be correct.

The job of a tool for *T* is to return correct answers and to avoid returning incorrect answers.

Universe of an RE Tool



Adopting IR Methods

RE field has often adopted (and adapted) IR algorithms to develop tools for NL RE tasks.

Quite naturally RE field has adopted also IR's measures:

- **precision, *P*,**
- **recall, *R*, and**
- **the *F*-measure**

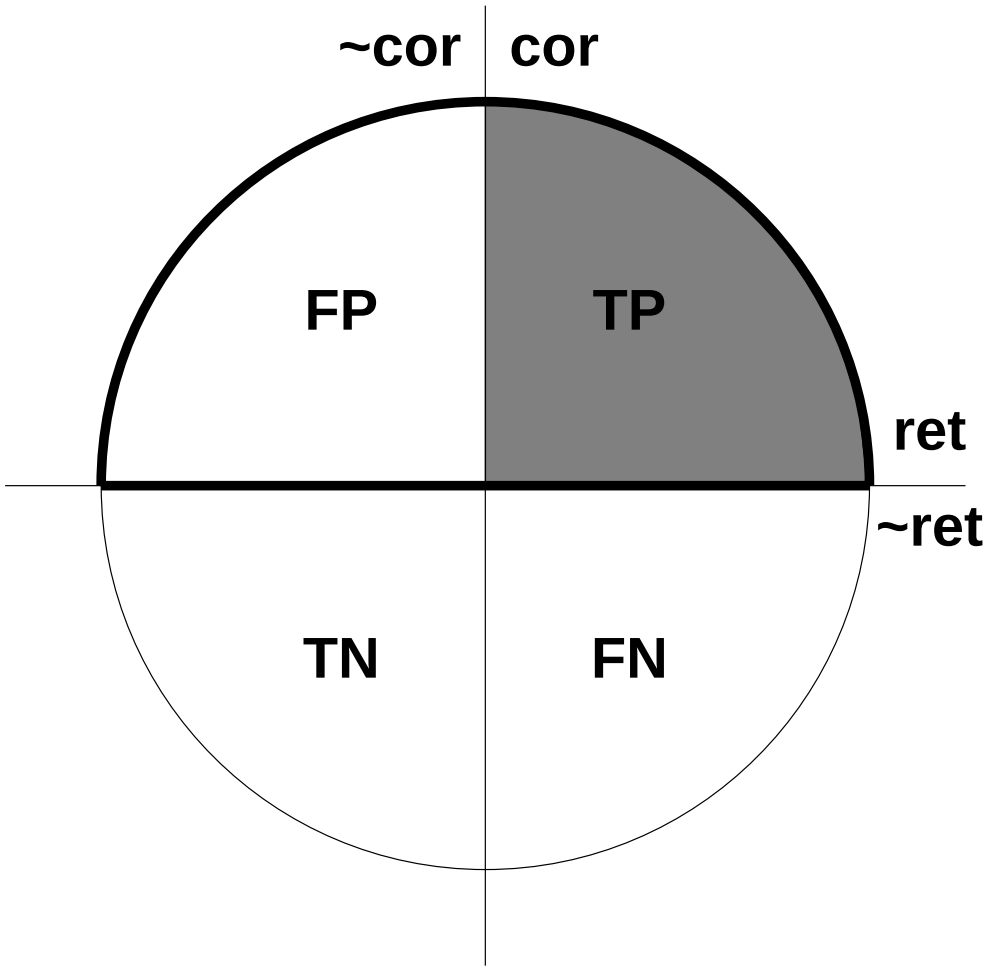
Precision

***P* is the percentage of the tool-returned answers that are correct.**

$$P = \frac{|\ ret \cap cor\ |}{|\ ret\ |}$$

$$= \frac{|\ TP\ |}{|\ FP\ | + |\ TP\ |}$$

Precision



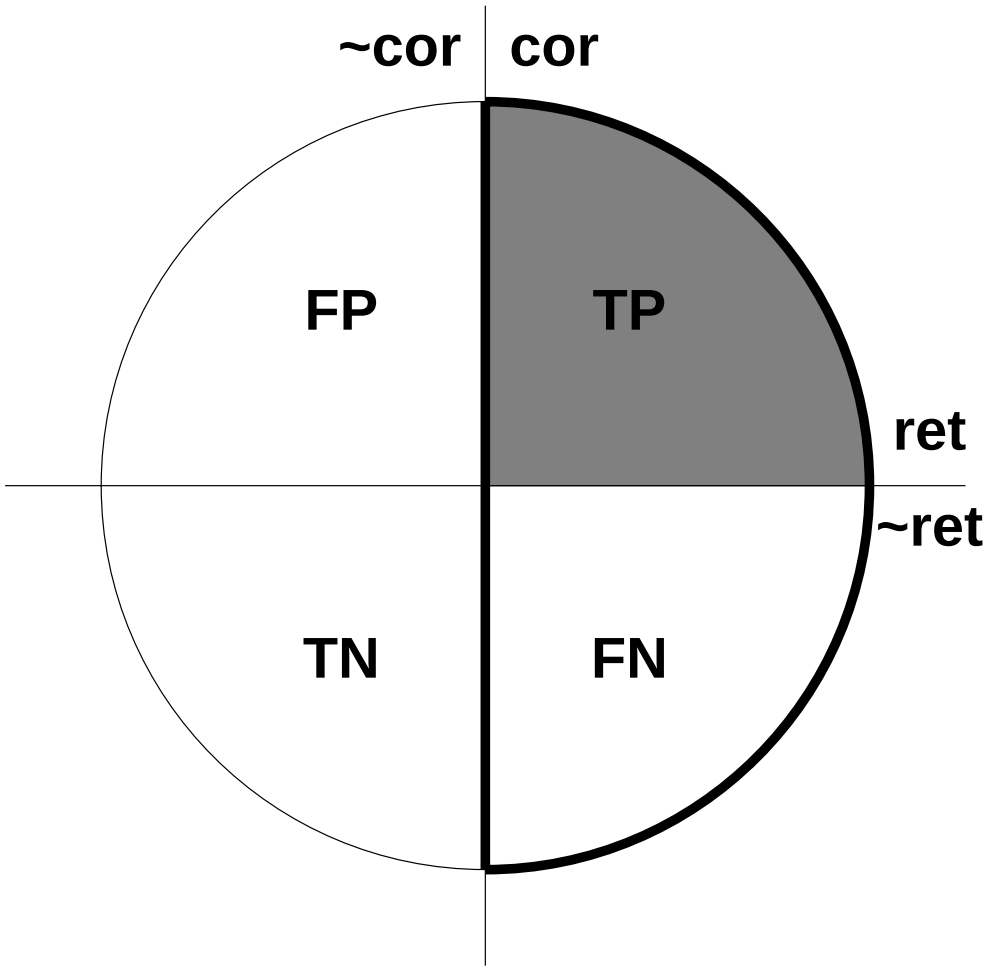
Recall

***R* is the percentage of the correct answers that the tool returns.**

$$R = \frac{|\ ret \cap cor\ |}{|\ cor\ |}$$

$$= \frac{|\ TP\ |}{|\ TP\ | + |\ FN\ |}$$

Recall



F-Measure

F*-measure: harmonic mean of *P* and *R
(harmonic mean is the reciprocal of the arithmetic mean of the reciprocals)

Popularly used as a composite measure.

$$F = \frac{1}{\frac{\frac{1}{P} + \frac{1}{R}}{2}} = 2 \cdot \frac{P \cdot R}{P + R}$$

Weighted *F*-Measure

For situations in which *R* and *P* are not equally important, there is a weighted version of the *F*-measure:

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

Here, β is the ratio by which it is desired to weight *R* more than *P*.

Note That

$F = F_1$

As β grows, F_β approaches R (and P becomes irrelevant).

If Recall Very Very Important

Now, as $w \to \infty$,

$$F_w \approx w^2 \cdot \frac{P \cdot R}{w^2 \cdot P}$$

$$= \frac{w^2 \cdot P \cdot R}{w^2 \cdot P} = R$$

As the weight of *R* goes up, the F-measure begins to approximate simply *R* !

If Precision Very Very Important

Then, as $w \to 0$,

$$F_w \approx 1 \cdot \frac{P \cdot R}{R}$$

$$= P$$

which is what we expect.

R vs *P* Tradeoff

***P* and *R* can usually be traded off in an IR algorithm:**

- **increase *R* at the cost of decreasing *P*, or**
- **increase *P* at the cost of decreasing *R***

Extremes of Tradeoff

Extremes of this tradeoff are:

1. **tool returns all possible answers, correct and incorrect: for**

   $R = 100\%$, $P = C$,

   where $C = \frac{\#\ \text{correctAnswers}}{\#\ \text{answers}}$

2. **tool returns only one answer, a correct one: for**

   $P = 100\%$, $R = \varepsilon$,

   where $\varepsilon = \frac{1}{\#\ \text{correctAnswers}}$

Extremes are Useless

Extremes are useless, because in either case, …

the entire task must be done manually on the original document in order to find *exactly* the correct answers.

100% Recall Useless?

Returning everything to get 100% *R* doesn't save any real work, because we still have to manually search the entire document.

This is why we are wary of claims of 100% *R* … Maybe it's a case of this phenomenon!

What is missing?

Summarization

Summarization

If we can return a subdocument significantly smaller than the original …

that contains *all* relevant items, …

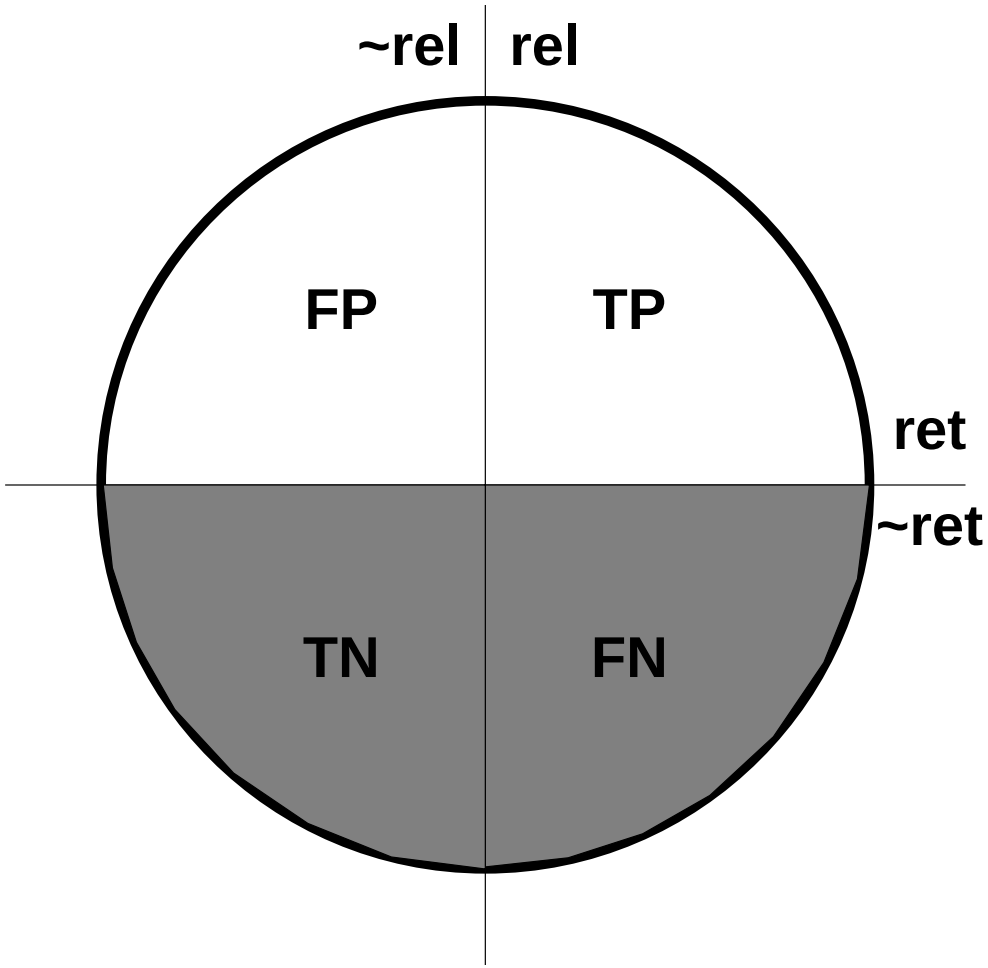
then we have saved some real work.

Summarization Measure

Summarization = fraction of the original document that is eliminated from the return

$$S = \frac{|\sim ret|}{|\sim ret \cup ret|} = \frac{|\sim ret|}{|\sim rel \cup rel|}$$

$$= \frac{|TN| + |FN|}{|TN| + |FN| + |TP| + |FP|}$$


~rel
rel
FP
TP
ret
~ret
TN
FN

How to Use Summarization

We would *love* a tool with 100% *R* and 90% *S*.

Then we really do not care about *P*.

In Other Words

That is, if we can get rid of 90% of the document with the assurance that …
what is gotten rid of contains *only irrelevant* items and thus …

what is returned contains *all* the relevant items, …

then we are *very happy*! ☺

Historically, IR Tasks

IR field, e.g., for search engine task, values *P* higher than *R*:

Valueing P more than R

Makes sense:

Search for a Portuguese restaurant.

All you need is 1 correct answer:

$$R = \frac{1}{\#\ a\text{correctAnswers}}$$

But you are *very* annoyed at having to wade through many FPs to get to the 1 correct answer, i.e.,

with low *P*

NL RE Task

Very different from IR task:

- **task is hairy, and**
- **often critical to find all correct answers, for *R* = 100%, e.g. for a safety- or security-critical CBS.**

Hairy Task

On small scale, finding a correct answer in a single document, a hairy NL RE task, …

e.g., deciding whether a particular sentence in one RS has a defect, …

is easy.

Hairy Task, Cont'd

However, in the context of typical large collection of large NL documents accompanying the development of a CBS, the hairy NL RE task, …

e.g., finding in all NL RSs for the CBS, all defects, …

some of which involve multiple sentences in multiple RSs, …

becomes *unmanageable*.

Hairy Task, Cont'd

It is the problem of finding *all* of the *few* matching pairs of needles distributed throughout multiple haystack.

“Hairy Task”?

Theorems, i.e., verification conditions, for proving a program consistent with its formal spec, are not particularly deep, …

involve high school algebra, …

but are incredibly messy, even unmanageable, requiring facts from all over the program and the proofs so far …

and require the help of a theorem proving tool.

We used to call these “hairy theorems”.

“Hairy Task”?, Cont’d

At one place I consulted, its interactive theorem prover was nicknamed “Hairy Reasoner” ☺ (with apologies to the late Harry Reasoner of ABC and CBS News)

Other more conventional words such as “complex” have their own baggage.

Hairiness Needs Tools

The very hairiness of a HT is what motivates us to develop tools to assist in performing the HT, …

particularly when, e.g. for safety- or security-critical CBS, …

***all* correct answers, …**

e.g., ambiguities, defects, or traces …

***must* be found.**

Hairiness Needs Tools, Cont'd

For such a tool, …

***R* is going to be more important than *P*, and …**

β in F_β will be > 1

What Affects R vs. P Tradeoff?

Three partially competing factors affecting relative importance of *R* and *P* are:

- **the value of β as a ratio of two time durations,**
- **the real-life cost of a failure to find a TP, and**
- **the real-life cost of FPs.**

Value of β

The value of β can be taken as ratio of

the time for a human to find a TP in a document

over

the time for a human to reject a tool-presented FP.

We will see how to get estimates during gold-standard construction.

Some Values of β

The panel paper gives some β values

ranging from 1.07 to 73.60 for the tasks:

predicting app ratings, estimating user experiences, & finding feature requests from app reviews;

finding ambiguities; and

finding trace links.

Gold Standard for *T*

Need a representative same document *D* for which a group *G* of humans have done *T* manually to obtain a list *L* of correct answers for *T* on *D*.

This list *L is* the gold standard.

***L* is used to measure *R* and *P* for any tool *t*, by comparing *t*'s output on *D* with *L*.**

Gather Data During *L*'s Construction

During *L*'s construction, gather following data

- **average time for anyone to find any correct answer = β's numerator,**
- **average time to decide the correctness of any potential answer = lower upper bound estimate for β's denominator, independent of any tool's actual value,**

During *L*'s Construction, Con't

- **average *R* of any human in *G*, relative to final *L* = estimate for humanly achievable high recall (HAHR).**

Real-life cost of not finding a TP

For a safety-critical CBS, this cost can include loss of life.

For a security-critical CBS, this cost can include loss of data.

Real-life cost of FPs

High annoyance with a tool's many FPs can deter the tool's use.

Tool vs. Manual

Should we use a tool for a particular HT *T*?

Have to compare tool's *R* with that of humans manually performing the *T* on the same documents.

Goal of 100% *R* ?

For a use of the HT in the development of a safety- or security-critical CBS, we need the tool to achieve *R* close to 100%.

Goal of 100% *R*, Cont'd

However,

- **achieving *R* = 100% for *T* is probably impossible, even for a human!**
- **there's no way to be sure that a tool or person has achieved *R* = 100% because the only way to measure *R* is to compare the tool or person's output with the set of all correct answers, which is impossible to obtain!**

Reality

For any task *T*, we aim to build a tool whose *R* beats that of a human manually performing *T*, i.e., the HAHR determined during gold standard construction.

Summary

To evaluate a tool *t* for a task *T*, we need

- **to have effective empirical ways to measure tool's and humans' *R* and *P*, and times to do *T*,**
- **to take into account the value of β and the real-life costs, and**
- **to compare tool's *R* and *P* and humans' *R* and *P* on the same set of documents.**