# Automation of Summary Evaluation by the Pyramid Method

Aaron Harnly

# Outline

A. The Problem: Summary Evaluation
   1. Approaches
   2. Intrinsic: intuitions
   3. Challenges
   4. DUC
   5. ROUGE

B. The Pyramid Method
   1. Motivation
   2. Algorithm

C. Automation
   1. Motivation
   2. Algorithms
   3. Results so far

D. Thoughts & Future Work

# A.1. Approaches

## Extrinsic

*evaluate the utility of a summary in the performance of a task*

- ☑ gold standard
- ☒ requires human subjects
- ☒ difficult
- ☒ time-consuming
- ☒ expensive

## Intrinsic

*judge quality of summary directly, based on analysis by some set of norms*

- ☑ promises generality
- ☑ can offer automation
- ☒ might not apply well
- ? requires good "norm"!
- ? which measure?

# A.2: Intrinsic Evaluation – intuitions

- What characteristics do we seek in a summary?

$$\frac{\text{faithfulness}}{\text{compactness}} \Rightarrow \text{low fidelity:}$$

*{1,2}-grams; paraphrase & synonymy*

$$\frac{\text{precision}}{\text{recall}} \Rightarrow \text{coverage-based measures:}$$

*how many of the {sentences, words, ideas}*

*from the model are found in the target?*

# A.2: Intrinsic Evaluation – intuitions

- Approaches to measuring content coverage:
  - manual v. automatic
  - sentence co-selection
    - *recall, kappa, sentence-rank, relative utility*
    - **pros:** easy to include an "importance" measure
    - **cons:** extractive only; variation in focus
  - content-based similarity
    - *n-gram overlap, LCS, cosine*
    - **pros:** finer-grained; easy to automate
    - **cons:** synonymy, variation in focus
  - human-judged similarity
    - **pros:** overcomes challenges of synonymy
    - **cons:** reliability

# A.3: Challenges

- No single perfect summary
  - reasonable summaries can differ in *focus*
  - strategies:
    - build a single template from multiple reference summaries
    - somehow account for "equally-good" content?
- Content judgments
  - disagreement by judges: how well does the target summary cover the model summary?
  - strategies:
    - oh well, just do it anyway! ;)
- Score Stability
  - How many {reference, test} summaries are required to reliably distinguish systems?

# A.4: DUC Procedure

1. Human creates a model summary
2. Model summary is split into units (roughly clauses or EDUs)
3. Target summary is split into sentences
4. For each model unit:
   a. find all target units expressing at least some facts from this model unit
   b. assess: these target units, as a group, express x% of the meaning expressed by the model unit
5. Final score = average score across all model content units

# A.4: DUC Procedure – Limitations

- Subjective assessment of "meaning coverage"
  - Lin and Hovy 2002: Judges given the same model unit and same target unit assigned identical score only 82% of time
  - > 4% had three different scores
- Single model
  - single reference summary means target summaries will be punished or rewarded by chance correspondence with model
  - experimental choice of different model causes average of {43%, 69%} change in absolute score; but over 20+ docsets, system rankings stable
- No provision for relative importance of information from target summary

# A.5: ROUGE

- ROUGE
  - a bevy of automatic content overlap-based methods
    - built by analogy, of course, to *BLEU*
    - n-gram co-occurrence; LCS; W-LCS; skip-bigram;
  - NB that some of these measures implicitly give higher scores to summaries that contain text-chunks present in multiple reference summaries
  - Shown to correlate well with DUC manual method given > 30 single-docsets, or > 4 multi-docsets
  - Multiple references may stabilize scores sooner, but going from 1->2 actually destabilizes in some cases
  - Q: Is there any reason to prefer fewer, multi-ref docsets vs. more, single-ref docsets?

# B.1: The Pyramid Method

- Designed to capture two characteristics of summarization:
  - two summaries with different content can be equally 'good'
  - some content is more important
- Essential idea:
  - Explicitly assume multiple ref's are needed
  - Find sets of text fragments in different summaries that express approximately the same meaning
  - Use frequency as a marker of importance
  - Give higher score to summaries containing more important content

# B.2: Summary Content Units

An SCU is a *set* of contributors that express the same meaning

A

In 1998 two Libyans indicted in 1991 for the Lockerbie bombing were still in Libya.

C

Two Libyans were indicted in 1991 for blowing up a Pan Am jumbo jet over Lockerbie, Scotland in 1988.

J

A ten-year deadlock over trying two Libyans linked to the Lockerbie bombing appears close to a conclusion.

SCU #1

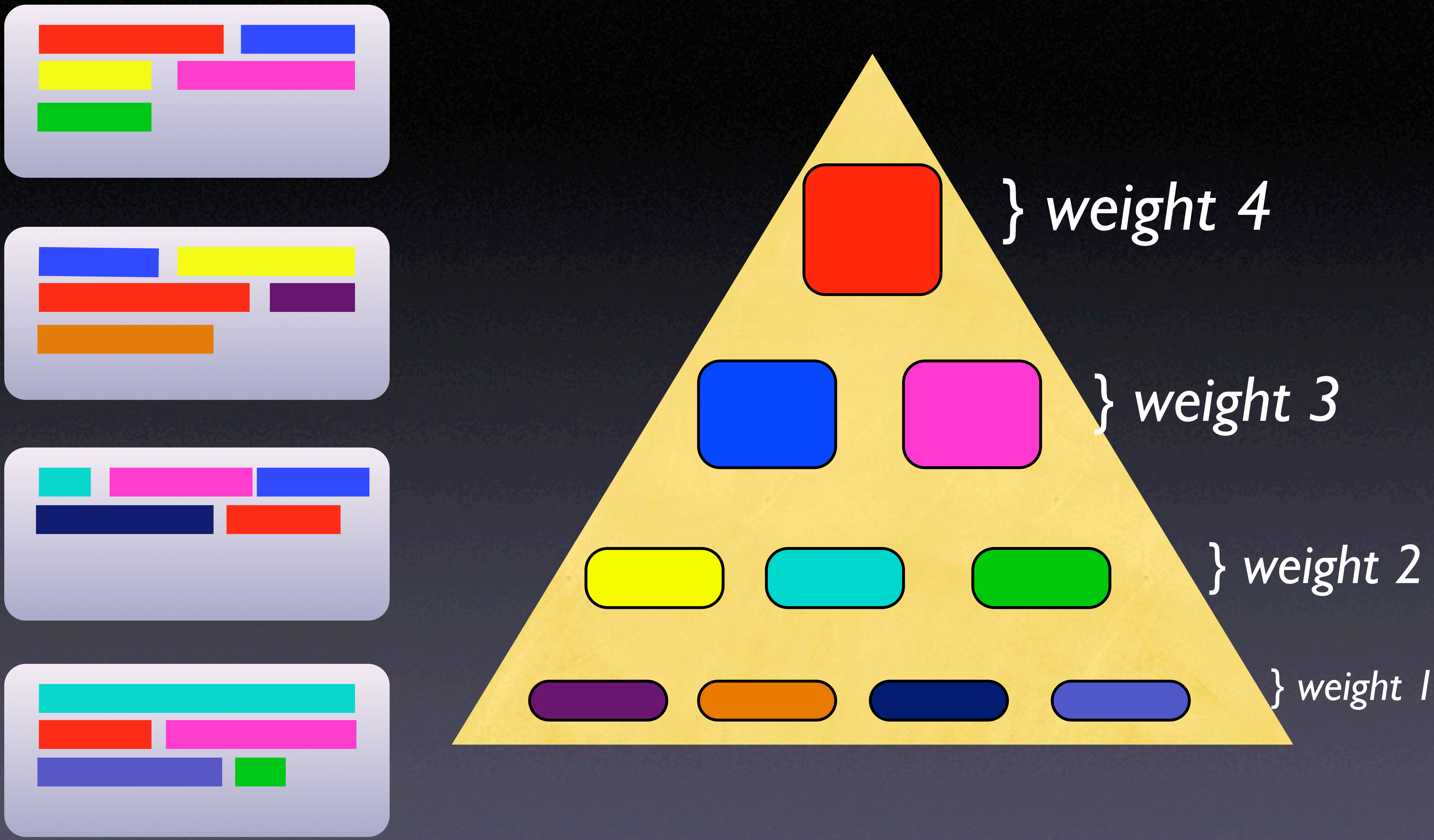
"The crime in question was the Lockerbie, Scotland bombing"

A: for the Lockerbie bombing

C: for blowing up ... over Lockerbie, Scotland

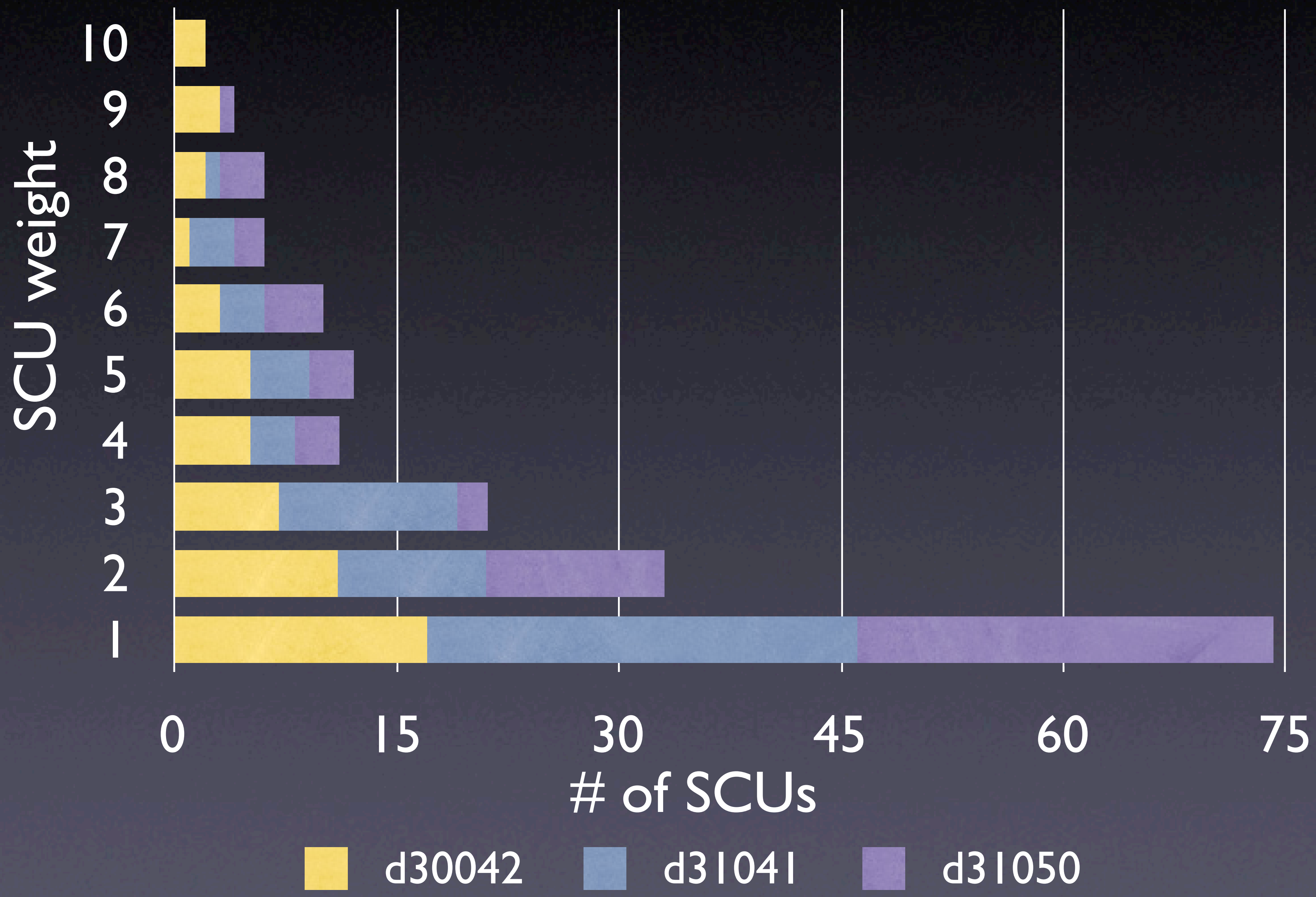
J: linked to the Lockerbie bombing

# B.2: Building the Pyramid



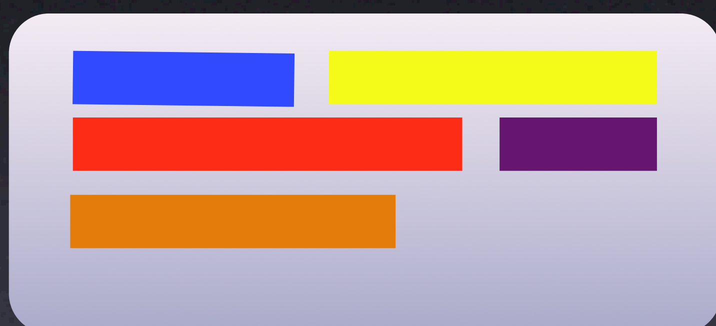
# B.2: Building the Pyramid

- How “pyramidal” are pyramids, anyway?

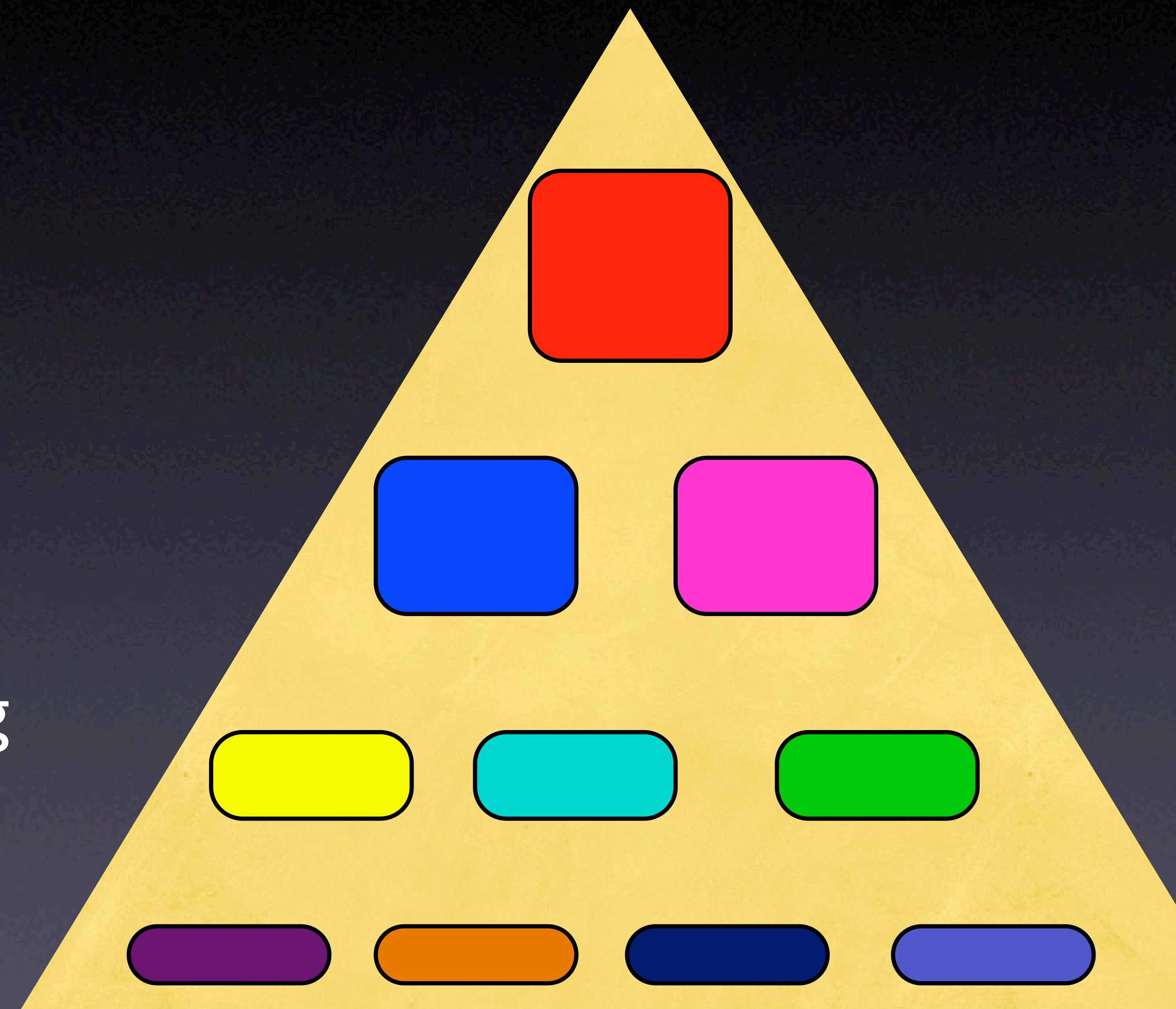


# B.2: Scoring new summaries

Task: exhaustively assign the text of the summary to extant SCUs



(But text expressing meaning not already in the pyramid can be assigned to new "singleton" SCUs)



## B.2: Scoring new summaries

- Total Pyramid score is:

$$\text{ratio of } \frac{\text{sum of weights of SCUs in target}}{\text{sum of weights of an optimal summary with same \# of SCUs}}$$

$$\text{or: } \frac{\text{\# of model contributors with paraphrase in target}}{\text{max possible with same \# of target contributors}}$$

# B.3: Pyramid Method – Thoughts

- Comparison to multi-ref DUC
  - how much would DUC improve with multiple reference summaries?
  - What would Pyramid do differently?
    - finer-grained chunking
    - is "means about the same" a more reliable criterion than "covers about x% of the meaning?"

# C.1: Automating the Pyramid Method

- Pyramid method has two main tasks:
  - 1. Building the pyramid
  - 2. Scoring new target summaries
- We have focused on task #2 for now

# C.2: Outline of Algorithms

- Task: exhaustively assign the text of an incoming target summary to the extant SCUs of a pyramid
- Outline of procedure:
  - a. Enumerate all possible contributors.
  - b. Match each possible contributor to the SCU(s) expressing similar meaning
  - c. Choose a covering, disjoint set of possible contributors.

# C.2: Algorithms - a

- a: Enumeration of possible contributors (with new constraint: contiguous)
  - *simply* $\frac{n(n+1)}{2}$ *contiguous contributors*

```
In | 1998 | two | Libyans | indicted | in | 1991 | for | the | Lockerbie | bombing

In___1998 | two___Libyans | indicted___in | 1991___for | the___Lockerbie | bombing
In | 1998___two | Libyans___indicted | in___1991 | for___the | Lockerbie___bombing

In___1998___two | Libyans___indicted___in | 1991___for___the | Lockerbie   bombing
In | 1998___two___Libyans | indicted___in___1991 | for___the___Lockerbie | bombing
In   1998 | two___Libyans___indicted | in___1991___for | the___Lockerbie___bombing

In___1998___two___Libyans | indicted___in___1991___for | the   Lockerbie   bombing
In | 1998___two___Libyans___indicted | in___1991___for___the | Lockerbie   bombing
In   1998 | two___Libyans___indicted___in | 1991___for___the___Lockerbie | bombing
In   1998   two | Libyans___indicted___in___1991 | for___the___Lockerbie___bombing

                                  etc.
```

# C.2: Algorithms – b

b. Match each possible contributor to SCU(s)

- b.  Match each possible contributor to the SCU(s) expressing similar meaning

- This means we need a similarity metric between *contributors* and *sets of contributors*

- Essentially a problem of cluster pairs:
  - *single link: max* of pairwise similarity
  - *average link: mean* of pairwise similarity
  - *complete link: min* of pairwise similarity
  - similarity to a *template*
  - multiple sequence alignment

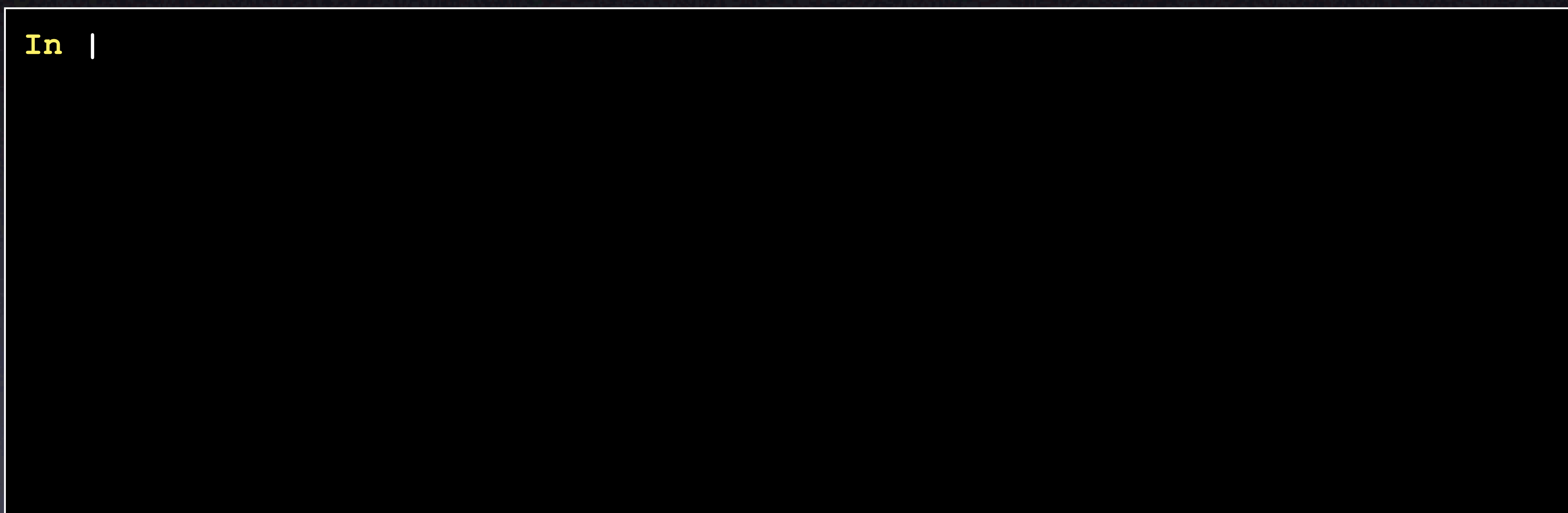
# C.2: Algorithms – b

b. Match each possible contributor to SCU(s)

- So, we first need a pairwise similarity metric
- Again, many possibilities:
  - string edit distance
  - ngram overlap
  - centroid
  - SIMFINDER
  - tree edit distance of dependency parse?

# C.2: Algorithms – c

- c. Choose a covering, disjoint set of possible contributors.



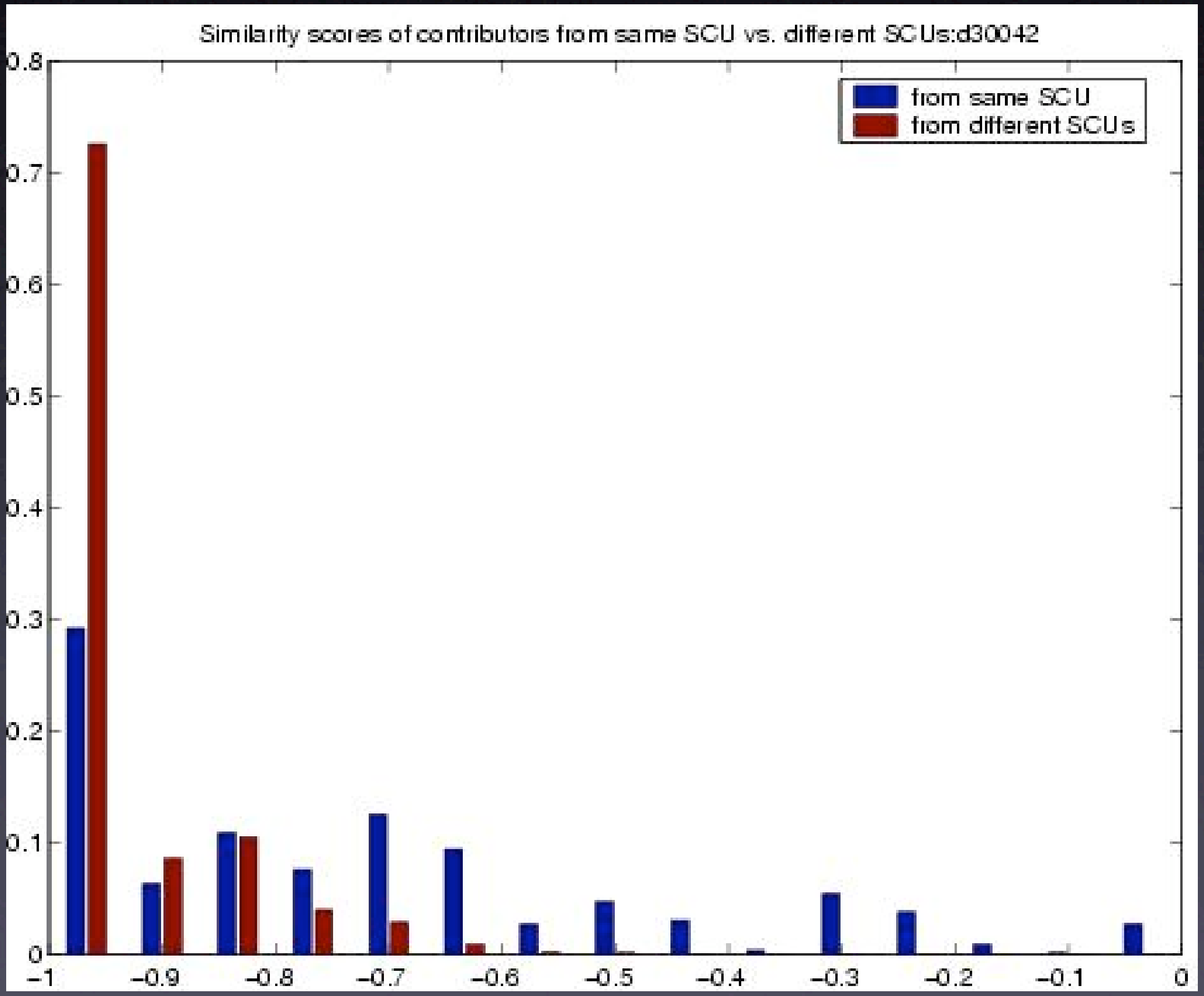

- obvious answer: a DP algorithm selecting the best contributor set for the first *i* words
  - but beware of constraint relaxations!

# Automating the Pyramid Method: Initial Results

- 2. Selection of pairwise similarity metric
  - initial trials:
    - string edit distance
    - ngram overlap
  - *a great pairwise similarity metric should cleanly separate contributors **known** to be in the same SCU from those **known** to be in different SCUs*

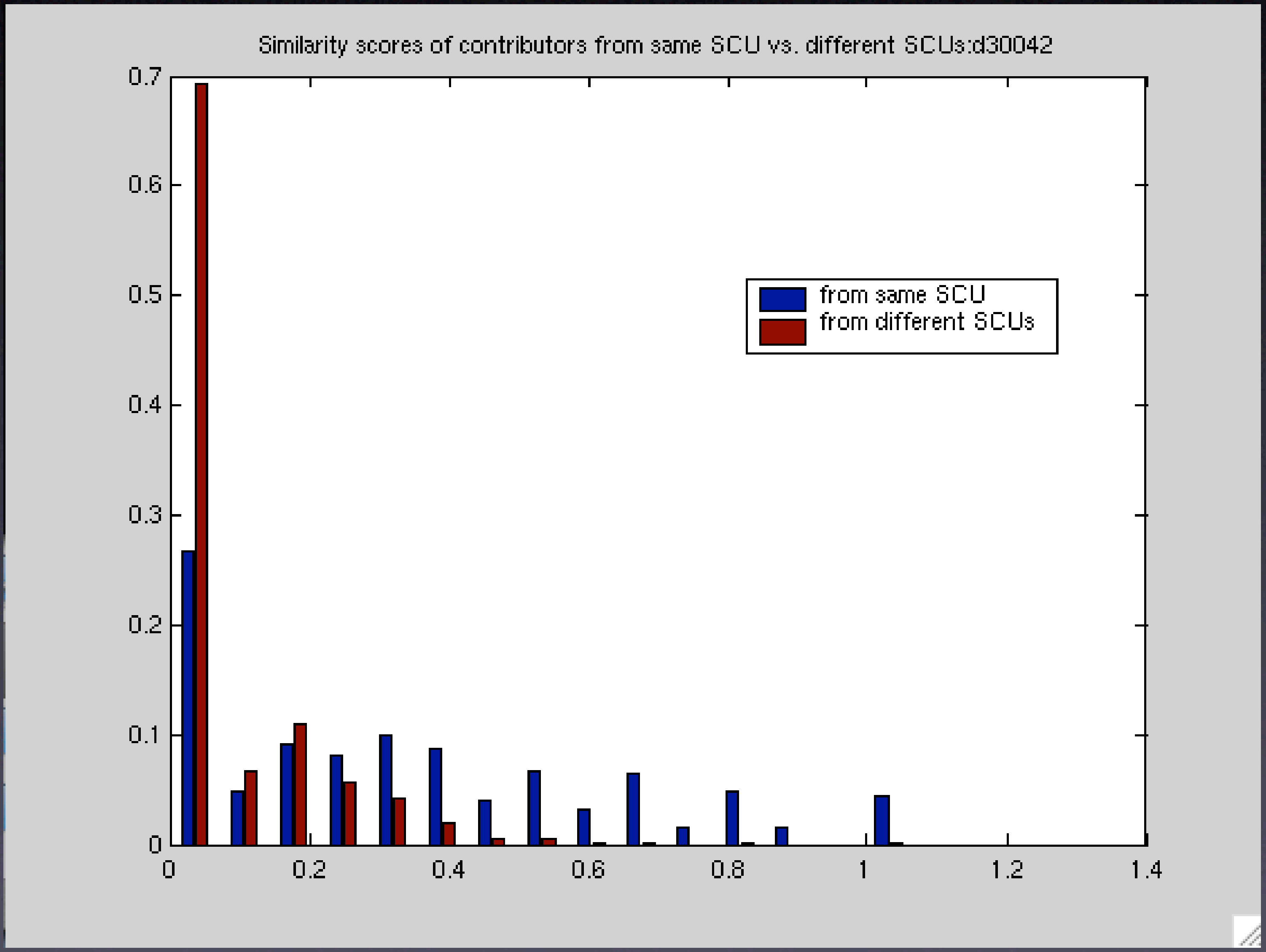
# C.3: Automation – Initial Results

- 2. Selection of pairwise similarity metric:
  string edit distance



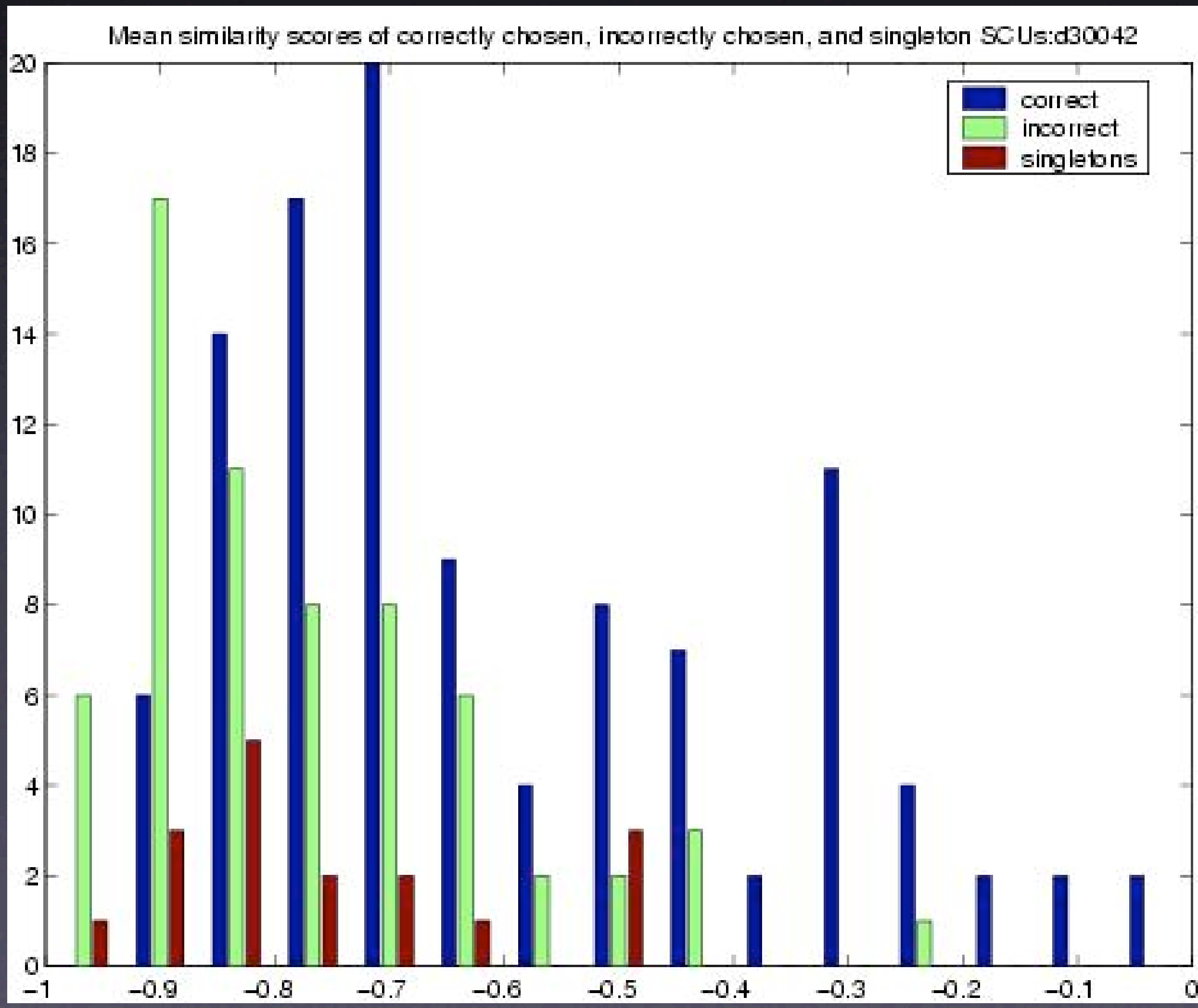
# C.3: Automation – Initial Results

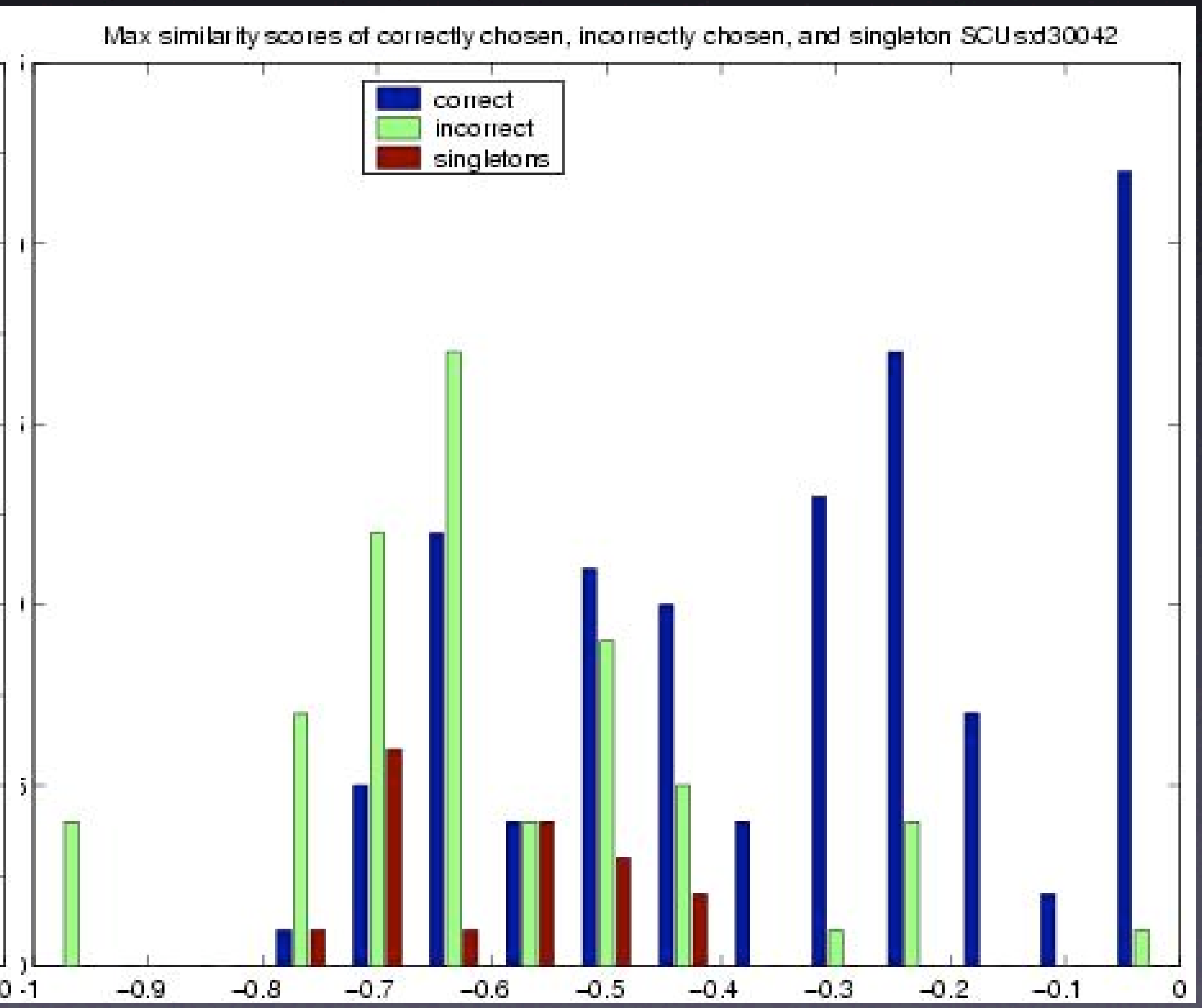
- 2. Selection of pairwise similarity metric: word overlap



# C.3: Automation – Initial Results

- 2. Selection of clustering method:
  similarity of single contributor to set




average-link (*mean*) single-link (*max*)

# C.3: Automation – Initial Results

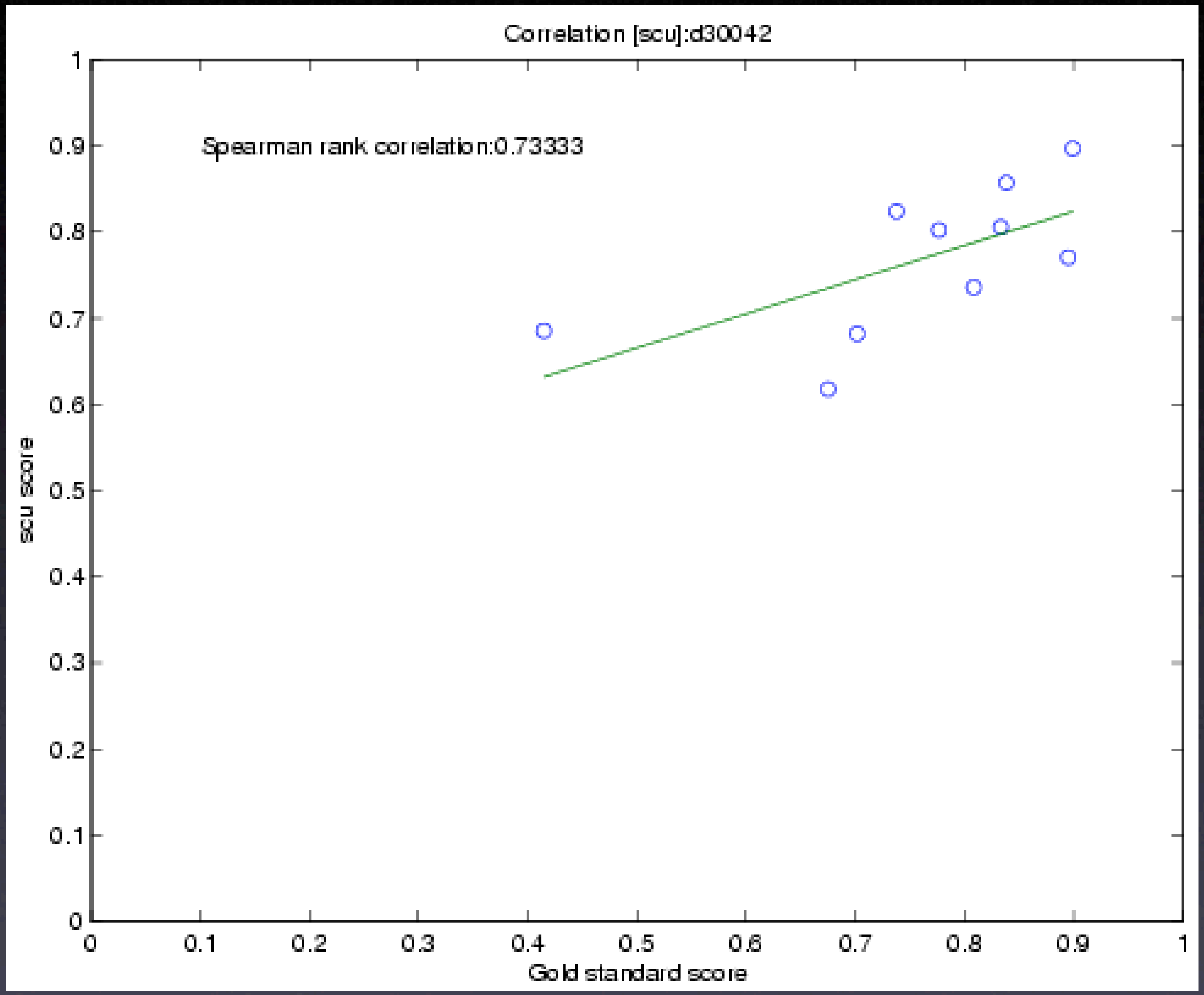
- Putting it all together:
  *with string-edit-distance, single-link similarity metric*

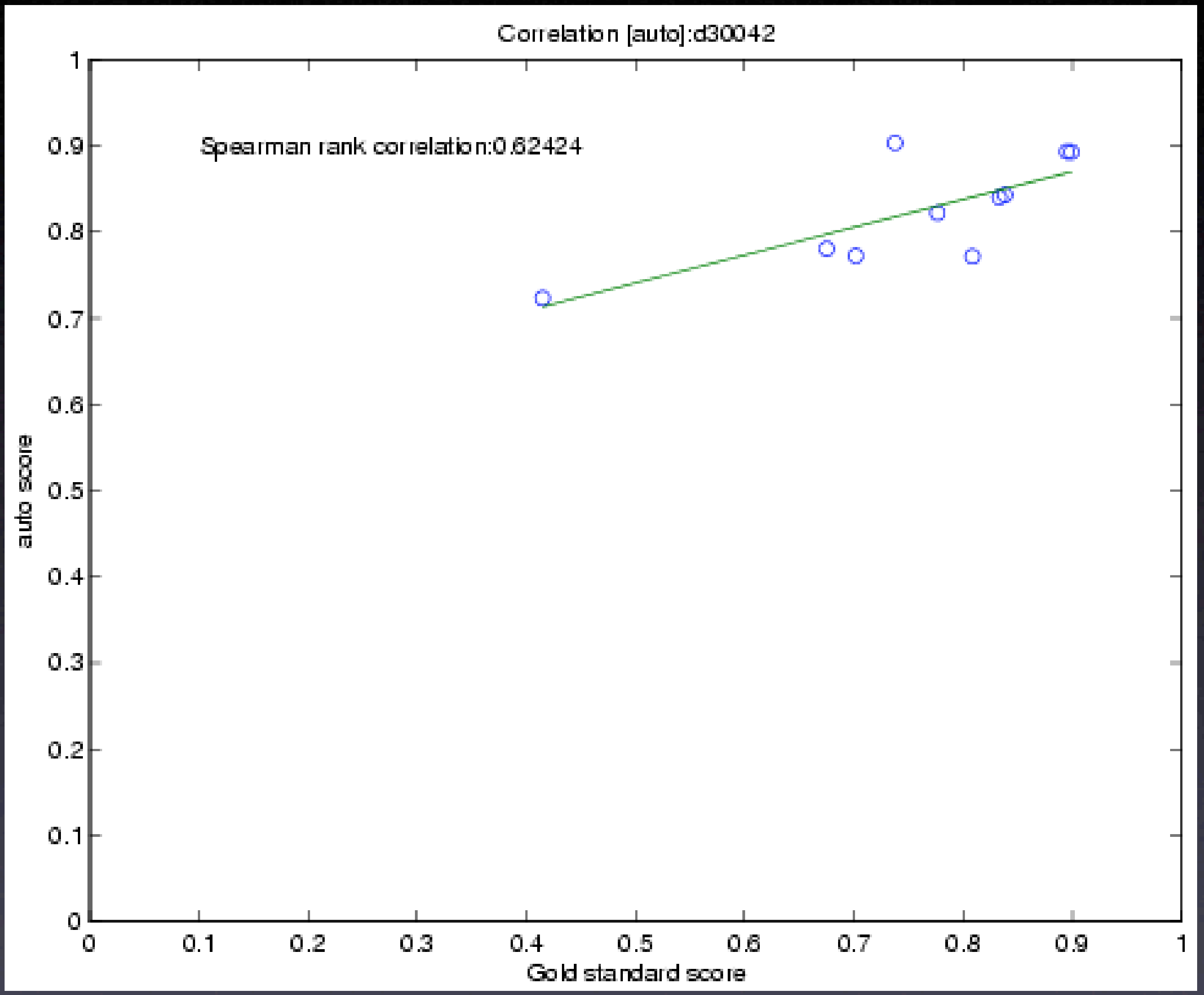
- Evaluation:
  - *n*-fold cross validation: hold out one summary at a time; score it against pyramid built with the rest of the summaries

  - Spearman's rank correlation to the human-annotated pyramid scores

# C.3: Automation – Initial Results

## Two levels of automation:




* hand-annotated contributor selection, automatic SCU assignment

* automatic contributor selection + SCU assignment

# D.1: Lots to do!

- Lots of work to be done!
- Similarity metrics
  - other surface string pairwise metrics
  - explore interaction with clustering method
- SCU selection
  - right now we assign each contributor to its "best fit" SCU
  - but perhaps allowing n-bests would give the DP contributor selection more flexibility?

# D.1: Lots to do!

- More data
  - <u>need</u> to test this across many more docsets
  - Dave E. is annotating more pyramids
- Try full automation: pyramid-building
  - clustering possible contributors
  - should try it and see what comes out!

# Questions / Comments?

-