`jlink`, `jsort` and `jgrep`: Record Linkage, Text Scaling and Classification with a Decision Model*

Khaled Eltokhy
CUNY Graduate Center

September 23, 2026

Keywords: record linkage; text as data; decision models; innovation data; replication.
JEL codes: C81; C55; O30.

1 Introduction

Innovation research often relies on text to construct many of its variables. For instance, patents have to be matched with firms based on assignee names that rarely correspond exactly to the firm's official records (Hall, Jaffe and Trajtenberg, 2001). The content of patents and official statements is now measured directly from their text (Gentzkow, Kelly and Taddy, 2019; Kelly et al., 2021). In practice, researchers usually try to match strings (typically with some manual review; Abramitzky et al., 2021), build lists of keywords, or train classifiers on data where observations have been labeled by hand. All three economize on the scarce input in this work, a person's judgment about what a text says. String similarity stands in for a reader deciding whether two names belong to the same firm, a keyword list stands in for a reader deciding what a document is about, and a classifier extends a small set of hand labels to the rest of the corpus. Each substitute is used because judgment is slow and expensive, and each changes what is measured: the variable becomes whatever the threshold, the word list or the labels happen to capture, and the reader of the paper sees the substitute rather than the question it replaced.

A decision model makes that judgment cheap. It answers a question written as an ordinary sentence and returns a probability, quickly and cheaply enough that the question can be asked of every record rather than a sample, so the substitutes no longer have to decide. I can ask the question I mean, change its definition and rerun the whole dataset in minutes, and audit a stratified sample to learn how often the answers are right. The three tools I describe are built on this: jlink, for record linkage; jsort, for scaling texts from pairwise comparisons; and jgrep, for classification by description. On public benchmarks, without training data, jlink beats tuned string similarity on four of five linkage tasks, jsort beats the best readability formula, and jgrep beats a keyword pattern.

*Extended abstract prepared for the NBER Innovation Information Initiative Technical Working Group, Cambridge, MA, December 4–5, 2026. Software: `jlink`, `jsort`, `jgrep` and `jevkit-runtime`, MIT licensed, on GitHub and PyPI.

2 Design

Common structure. All three tools turn the task of making measurements into questions about text, which I write in plain English as complete sentences, and a decision model (Jev, from TypeSafe) returns the probability that the sentence is true for a given piece of text. This takes about a fifth of a second and costs about a thousandth of a cent. The sentence that defines the variable then goes into the paper as part of the description of my method, where a reader can inspect or change it. Every answer I get is stored along with the model, the original text, and the question that gave rise to it. So if I rerun the analysis, I will get exactly the same numbers again. And the same requests can be sent to an open model running on my own machine when the data cannot leave it.

Record linkage (jlink). jlink is a tool for linking two tables of records. In the first stage, it proposes candidate pairs of records from the two datasets, on my own machine, using overlapping strings of characters, initials, or exact values for fields or windows over dates and numbers. In the second stage, each proposed pair is sent to Jev for a judgment of whether it should be linked under some rule I write as a sentence, which can involve complex decisions such as ignoring legal-form suffixes and counting a subsidiary company as part of its parent. Finally, when all pairs have been judged, jlink resolves the set of pairs into links between records in the datasets, either one-to-one or many-to-one, at a threshold I choose. To validate any particular linkage produced by jlink, I take a sample of the records, stratified by their estimated probability of being linked, and label by hand whether pairs of records in the sample are linked. These labels then provide estimates of precision, recall, and calibration, along with confidence intervals. jlink also writes out a paragraph describing what I did in methods section format, quoting the rule verbatim, and can be run from Python, the command line, Stata and R (Listing 1).

Listing 1: One command per task.

```
. jlink using patents.dta, on(conm=assignee state) entity(firm) ///
      define("A subsidiary or division counts as its parent company.") ///
      leftid(gvkey) rightid(assignee_id) merge
$ jsort --whole -o "more hawkish about inflation" statements/*.txt
$ jgrep -o "an unsolicited spam, scam or marketing text message" sms.txt
```

Scaling (jsort). With the second tool, jsort, I ask Jev to tell me the probability that one of two texts is ranked higher than the other on some scale I describe. Each text takes part in about ten comparisons. Then, I use these probabilities together with a Bradley-Terry model (Bradley and Terry, 1952) estimated by fractional logit, which gives me a score (and a robust standard error) for every text.

Classification (jgrep). The simplest of the tools is called jgrep. It asks, for each line or document in a file, whether this text matches the description I have written, and returns the probability that it does. I can use the result to filter through a file, in which case it works like grep, except that instead of giving it a pattern to match, I give it a textual description of what I'm looking for. Alternatively, I can use its output as another variable whose value depends on the text; I can ask several different questions about the same text in the same call.

3 Validation

Linkage benchmarks. I ran jlink on the five benchmarks with known matches, using the default threshold of 0.5 without tuning it on the answers. Each string-similarity baseline instead got the single threshold that maximizes its F1. On four out of five I get a higher F1 than any string similarity measure (Table 1, panel A), including in the case of matching patent assignees to firms in Compustat (where jlink obtains an F1 score of 0.73 against 0.69 for string similarity). However, there is one benchmark where string similarity outperforms jlink, which is a set of synthetic person records that only vary by typos, the case string similarity was designed for. In addition, without having seen any labels, jlink outperformed LinkTransformer's pretrained models (Arora and Dell, 2023) on eleven out of twelve standard benchmarks. When fine-tuned on a few hundred labelled matches from each benchmark, LinkTransformer outperformed jlink on eight of them.

Definition dependence. The most important test for economists is not how well the method does on a given definition, but whether its answers change when the definition changes. To test this, I constructed two small data sets by hand from publicly available company filings, in which the same pairs of companies were judged to match or not under four different definitions: legal entity, corporate group, physical site and operating business. In the first set, which had 40 judgments, jlink got 39 right. In the second set (33 judgments), jlink got 30 right. Of course, a similarity score gives just one number for any pair of firms, whatever the definition. No threshold on any such score could do better than getting 30 out of 40 judgments right, or 23 out of 33. This matters for innovation data, where whether a subsidiary's patents count toward its parent is a research choice; with jlink the choice is written down, and changing it changes the linkage.

Scaling. The CommonLit corpus of readability excerpts (Crossley et al., 2023) provides a direct test of jsort. The corpus was built the way jsort works: each teacher was shown pairs of texts, and asked to select whichever they thought would be easier for students to read. On a set of 300 excerpts from the corpus, I find that the score given by jsort with the description "easier to read" correlates 0.82 with the scale implied by the teachers' responses. The extent to which the teachers disagreed sets an upper bound on how well any measure could correlate with their scale, at around 0.88. By comparison, the best readability formula in the CommonLit corpus achieves only 0.66 (Table 1).

For a second test, I sorted the opening statements from the 95 Federal Open Market Committee press conferences since 2011 based on how hawkish they are about inflation. There were 32 meetings over this period when the Fed did move rates, and the rank correlation between a statement's score and the size of the rate change announced that day was 0.80. The correlation across all 95 meetings is lower, 0.46, simply because if the committee didn't move rates then the size of the move is zero, no matter what the chair said. On average, statements at meetings that held rates score between those at cuts and those at hikes. Joe Weisenthal, who runs an independent project called FedLock (Weisenthal, 2026) that ranks about 4,000 Fed speeches by their relative hawkishness or dovishness, has released two recent versions of his rankings. My rankings correlate 0.85 and 0.93 with those two releases, which themselves have a rank correlation of 0.88, so jsort agrees with FedLock about as closely as FedLock agrees with itself.

Classification and local inference. The classification task is to determine whether a given text message is "an unsolicited spam, scam or marketing text message". To do this I used the complete collection of 5,574 messages from the UCI machine learning repository (Almeida, Gómez Hidalgo and Yamakami, 2011) and asked Jev to classify each one. My score on that data was an F1 of 0.91, compared with 0.72 for a keyword pattern I wrote before seeing the data.

Finally, for data that cannot leave my machine, I ran the same linkage benchmarks on DiffusionGemma, an open model. On four of the five tasks, it came within 0.03 of Jev's F1 scores (Table 1, panel B). However, each sample took around a quarter of an hour to process, and a much smaller open model performed at near chance level on these tasks.

4 Limitations and Open Questions

The main limitation of Jev's probabilities is that they are not well calibrated. They rank true matches above false ones well, but pairs with probability 0.8 or higher on the firm benchmark were true matches 95-98% of the time, and pairs between 0.5 and 0.8 (average score 0.64) were only true 39% of the time. In other words, until my sample audit shows otherwise, I have to treat these numbers as an ordinal scale rather than a cardinal one.

jlink can only judge the pairs its first stage proposes: on the firm benchmark, that stage proposed 67 percent of the known links, and jlink found 93 percent of the true links among them; the misses include subsidiaries whose names share nothing with their parent's. The benchmarks are also public and may have been seen in training, and each result is a single run.

The main open question is how the probabilities should enter a regression. Calibrated probabilities carry Berkson-type error, uncorrelated with the probability itself, and so need not attenuate the estimate the way misclassified hard labels do. Jev's raw scores are not calibrated, so the question is whether scores recalibrated on the audit sample remove the bias or whether the usual generated-regressor corrections (Pagan, 1984) are still needed. Answering it requires a hand-labelled validation set, which I have not yet built.

References

Abramitzky, R., L. Boustan, K. Eriksson, J. Feigenbaum, and S. Pérez (2021). Automated linking of historical data. *Journal of Economic Literature* 59(3), 865–918.

Almeida, T. A., J. M. Gómez Hidalgo, and A. Yamakami (2011). Contributions to the study of SMS spam filtering: New collection and results. In *Proceedings of the 11th ACM Symposium on Document Engineering*, 259–262.

Arora, A. and M. Dell (2023). LinkTransformer: A unified package for record linkage with transformer language models. arXiv:2309.00789.

Bradley, R. A. and M. E. Terry (1952). Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika* 39(3/4), 324–345.

Crossley, S., A. Heintz, J. S. Choi, J. Batchelor, M. Karimi, and A. Malatinszky (2023). A large-scaled corpus for assessing text readability. *Behavior Research Methods* 55, 491–507.

Gentzkow, M., B. Kelly, and M. Taddy (2019). Text as data. *Journal of Economic Literature* 57(3), 535–574.

Hall, B. H., A. B. Jaffe, and M. Trajtenberg (2001). The NBER patent citation data file: Lessons, insights and methodological tools. NBER Working Paper 8498.

Kelly, B., D. Papanikolaou, A. Seru, and M. Taddy (2021). Measuring technological innovation over the long run. *American Economic Review: Insights* 3(3), 303–320.

Pagan, A. (1984). Econometric issues in the analysis of regressions with generated regressors. *International Economic Review* 25(1), 221–247.

Weisenthal, J. (2026). FedLock: FOMC hawkishness. `https://jnathan9.github.io/fedlock/`, accessed September 23, 2026.

Table 1: Benchmark results

A. Hosted Jev against the strongest simple alternative, full public benchmarks		
Benchmark	Jev toolkit	Alternative
Record linkage (jlink), F1		
NBER assignees to Compustat	**0.73**	string similarity, tuned: 0.69
DBLP to ACM (publications)	**0.996**	string similarity: 0.975
Abt to Buy (products)	**0.92**	string similarity: 0.86
Amazon to Google (software)	**0.66**	string similarity: 0.64
FEBRL4 (people, typos only)	0.96	TF-IDF cosine: **0.998**
Same firm pairs, four definitions	**39/40; 30/33**	any similarity score: $\leq$30; $\leq$23
12 DeepMatcher sets, no labels	**ahead on 11/12**	LinkTransformer, zero-shot
Scaling (jsort)		
CommonLit readability (300), r	**0.82** (max. 0.88)	formula 0.66; one question 0.75
FOMC statements vs. rate move	0.80 (32 moves); 0.46 (95)	
FOMC statements vs. FedLock	0.85 to 0.93	FedLock vs. itself: 0.88
Classification (jgrep)		
UCI SMS spam (5,574), F1	**0.91**	keyword pattern: 0.72

B. Open model on the user's own machine against hosted Jev, same inputs		
Benchmark	Hosted Jev	DiffusionGemma, local
Record linkage (jlink), F1, 300-record samples		
Firms	**0.71**	0.69
Publications	**0.99**	0.98
Products	**0.94**	0.91
Software	**0.67**	0.65
People, typos only	**0.97**	0.84
Scaling (jsort)		
Readability, r with teachers' scale	**0.82**	0.78
FOMC vs. rate move, 32 moves	0.80	**0.92**
FOMC vs. FedLock	**0.93**	0.91
Classification (jgrep), 2,000-item sample		
SMS spam, F1	**0.91**	0.87

Notes: Hosted Jev at default thresholds, nothing tuned on the answers. A: each string-similarity baseline gets the threshold that maximizes its F1; FedLock is Joe Weisenthal's pairwise tournament over about 4,000 Fed speeches judged by an open-weights model. B: same inputs for both models; local runs on an Apple M3 Ultra; a 421-million-parameter model (Laya) was near chance on linkage and sorting. Bold marks the better figure where two are compared.