1 MOTIVATION

We have shown in previous work that the fast Beta linkage framework tends to outperform standard Fellegi Sunter when each record in B has at most one match in A. However, significant problems arise when a record in B has multiple matching records in A. In particular, since matching probability is normalized among all records in A, matching probability is split among multiple records such that none of the matches achieves high enough posterior probability to be identified through the Bayes estimate. This amounts to a paradox: the more matches that record B_j has in A, the less likely the algorithm is to find a match.

We attempt to resolve this paradox by extending the fabl framework to handle these internal duplications. We introduce a modified Dirichlet process prior that allows each record in B to match to potentially multiple records in B. We emphasize that we are not interested in deduplication within either dataset for its own sake, but rather aim to conduct record linkage in light the problems posed by these duplications. For this reason, we do not enforce transitive closure throughout the Gibbs sampler as done in Marchant et al. (2021) and Aleshin-Guendel & Sadinle (2023), but rather assume it, and create a coherent set of linkages through post-processing. This allows for a record linkage model that is robust to internal duplications while maintaining the computational advantages of the original `fabl` model.

2 MULTIPLE MATCH

We provide updated notation to allow us to describe one record in B having multiple matches in A. Let Z_j be a vector containing the indices for all of the records in A that are a match with record B_j, and let $\mathcal{Z} = \{Z_j | j = 1, \ldots, n_B\}$ denote the set of all such vectors. Let $|Z_j| = \sum_{k=1}^{\infty} Z_j^k > 0$ denote the number of links associated with record B_j.

We can allow each record in B to match to multiple records in A through a Dirichlet process prior. Define a vector of probabilities $\boldsymbol{\pi} = (\pi_0, \ldots)$ where π_k is the probability that some record in B has exactly k matches in A. In implementation, we model each π_k as a

product of conditional probabilities: let η_k be the probability that some record in B has at least k matches, given that it has at least $k-1$ matches. This gives us the stick breaking representation

$$\pi_k = (1-\eta_{k+1}) \prod_{c=1}^{k} \eta_c, \tag{1}$$

where η_k are independent random variables from a Beta$(\alpha_\eta, \beta_\eta)$ distribution.

Similar to fabl, we adopt a prior specification on $\mathcal{Z}$ so that each matching Z_j of length $|Z_j| = k$ is equally likely. Formally, we have

$$\mathcal{L}(\mathcal{Z}, \boldsymbol{m}, \boldsymbol{u} \mid \gamma) = \prod_{i=1}^{n_A} \prod_{j=1}^{n_B} \prod_{f=1}^{F} \prod_{l=1}^{L_f} \left[m_{fl}^{I(i \in Z_j)} u_{fl}^{I(i \notin Z_j)} \right]^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)}, \tag{2a}$$

$$\boldsymbol{m}_f \sim \text{Dirichlet}(\alpha_{f1}, \ldots, \alpha_{fL_f}), \forall f = 1, \ldots, F, \tag{2b}$$

$$\boldsymbol{u}_f \sim \text{Dirichlet}(\beta_{f1}, \ldots, \beta_{fL_f}), \forall f = 1, \ldots, F, \tag{2c}$$

$$p(Z_j = q | \boldsymbol{\pi}) = \begin{cases} \frac{(n_A - k)!}{n_A!} \pi_k, & q \neq 0, \\ \pi_0, & q = 0; \end{cases} \tag{2d}$$

$$\pi_k = (1-\eta_{k+1}) \prod_{c=1}^{k} \eta_c, \tag{2e}$$

$$\eta_k \sim \text{Beta}(\alpha_\eta, \beta_\eta). \tag{2f}$$

2.1 Gibbs Sampler

Through MCMC, we are able to sequentially sample the components Z_j^k of the matching vector Z_j. In this context, we use the sequence of priors

$$p(Z_j^k | \eta_k) = \begin{cases} \frac{\eta_k}{n_j^k}, & z_j^k \in N_j^k, \\ 1 - \eta_k, & z_j = 0; \end{cases} \tag{3}$$

$$\eta_k \sim \text{Beta}(\alpha_\eta, \beta_\eta) \tag{4}$$

where N_j^k is the set of records in A that are available to be matched with B_j, and $n_j^k = |N_j^k| = n_A - (k-1)$ is the number of such records.

This sequence of priors leads to the sequence of posteriors

$$p(Z_j^k|\eta_k, \boldsymbol{m}, \boldsymbol{u}, \gamma) = \begin{cases} w_{ij}, & z_j^k \in N_j^k, \\ n_j^k \frac{1-\eta_k}{\eta_k}, & z_j = 0; \end{cases} \tag{5}$$

$$\eta_k \sim \text{Beta}(\alpha_\eta + n_k(\mathcal{Z}), \beta_\eta + n_{k-1}(\mathcal{Z}) - n_k(\mathcal{Z})) \tag{6}$$

where $n_k(\mathcal{Z}) = \sum_{j=1}^{n_B} I(|Z_j| \geq k)$ is the number of records in B that have at least k matches in A. Note that $n_0(\mathcal{Z}) = n_B$, and that for each k, we can view $n_{k-1}(\mathcal{Z})$ as a number of trials, and $n_k(\mathcal{Z})$ as a number of successes.

This specification induces an extension of `fabl` with an iterative matching phase. In each iteration of the Gibbs sampler, we sample an initial set of links using η_1. For each record in B that was found to have a link, we remove the linked record in A from consideration, and then sample another potential link with η_2. We continue, using η_k in the k^{th} matching step, until no new links are found, at which we point the matching phase terminates. The $\boldsymbol{\eta}, \boldsymbol{m}$, and $\boldsymbol{u}$ parameters are estimated based on all of the links identified. Crucially, there is no need to specify a maximum number of links per record, as this estimated through the model.

2.2 Variational Inference

There are several reasons why this prior would not be practical through variational inference.

- Through MCMC, we can sequentially sample all plausible matches for record B_j. In variational inference however, we need to estimate the match probability for the joint distribution of ever possible combination of matches.
- Without hashing, the number of possible combinations is $\binom{n_A}{k}$. With hashing, this number is $P^k/k!$ (I think), and would still require some operations at the scale of $\binom{n_A}{k}$ when conducting the actual hashing.
- However, this approach might work if using aggressive indexing/filtering.

In cases where we know at least one file is duplicate free, we can safely use `vabl`. However, if we want to be cautious about the potential for duplicates, the multiple match prior under `fabl` is a good option.

3 SIMULATIONS

Show accuracy under various settings of maximum cluster size for different algorithms. Recall will fall very quickly for `fabl`. Since we can't use Jaro, precision for `fastLink` will be poor. Multilink may be very dependent on correctly specifying K. This multiple match approach should remain strong in all settings.

4 CASE STUDIES

Provide an example of record pairs that fabl cannot identify as matches because of duplication. I know there are cases in NCVR.

REFERENCES

Aleshin-Guendel, S. & Sadinle, M. (2023), 'Multifile partitioning for record linkage and duplicate detection', *Journal of the American Statistical Association* **0**(0), 1–10.
URL: *https://doi.org/10.1080/01621459.2021.2013242*

Marchant, N. G., Kaplan, A., Elazar, D. N., Rubinstein, B. I. P. & Steorts, R. C. (2021), 'd-blink: Distributed end-to-end bayesian entity resolution', *Journal of Computational and Graphical Statistics* **30**(2), 406–421.
URL: *https://doi.org/10.1080/10618600.2020.1825451*

5 APPENDIX