

Probabilistic Record Linkage Using Pretrained Text Embeddings

Joseph T. Ornstein^{*†}

March 4, 2025

Abstract

Pretrained text embeddings are a fast and scalable method for determining whether two texts have similar meaning, capturing not only lexical similarity, but semantic similarity as well. In this paper, I show how to incorporate these measures into a probabilistic record linkage procedure that yields considerable improvements in both precision and recall over existing methods. The procedure even allows researchers to link datasets across different languages. I validate the approach with a series of political science applications, and provide open-source statistical software for researchers to efficiently implement the proposed method.

Keywords: Probabilistic Record Linkage, Fuzzy String Matching, Embeddings, Large Language Models, GPT-3, GPT-4, Active Learning, Text-As-Data

1 Introduction

Empirical social scientists frequently need to merge information from multiple datasets prior to conducting their analyses, but it is only in rare cases that two datasets contain a shared variable that unambiguously identifies which records belong to the same entity. In the absence of such exact matching variables, researchers must perform *fuzzy record linkage*—linking records based on some measure of similarity between variables. When working with text data, existing approaches commonly rely on *lexical* measures of string similarity (Jaro, 1989). These include “edit distance” measures (e.g. Jaro-Winkler and Levenshtein distance), string metrics that compare the frequency distributions of characters (e.g. cosine similarity), and set theoretic measures (e.g. Jaccard similarity), among many others. The most commonly used and cited fuzzy record linkage procedures in political science employ one or more of these metrics to capture the distance between pairs of records (Enamorado, Fifield and Imai, 2019; Kaufman and Klevs, 2022) .

^{*}Assistant Professor, Department of Political Science, University of Georgia, jornstein@uga.edu

[†]Thanks to Elise Blasingame, Jason Byers, Rob Carroll, David Cottrell, Ted Enamorado, Kosuke Imai, Aaron Kaufman, George Krause, Avital Livny, Isaac Mehlhaff, Brandon Stewart, Maria Tislenko, Garrett Vande Kamp, and participants at Polmeth 2024 and SPSA 2025 for their helpful comments and suggestions. The **fuzzylink** R package is available at <https://github.com/joeornstein/fuzzylink>.

Lexical similarity is a powerful tool for record linkage when datasets contain misspellings, typos, or other irregularities in data entry. But these measures have well-understood shortcomings, particularly in cases where lexically dissimilar strings can be used to represent the same entity. For example, the name “Patricia” is more lexically similar to “Patrick” than it is to its nickname “Trish”. Many record linkage problems that political scientists encounter have this property, in which semantically similar records can be represented by lexically dissimilar strings. Elected officials may be referenced by their legal name in one dataset and their nickname in another. An organization may be listed under its full name in one dataset and an acronym in another. For scholars of comparative and international politics, records may even appear in multiple languages. When faced with record linkage problems like these, a measure that captures not only the lexical similarity between strings, but their *semantic* similarity as well, would be highly desirable.

Fortunately, such measures have recently become widely available, thanks to rapid advances in large language models (LLMs) based on the transformer architecture (Vaswani et al., 2017). These models encode language using *text embeddings*, wherein each word is represented by a real-valued vector of numbers (Rodriguez and Spirling, 2022). Once trained, the distance between these text embeddings provides a useful measure of semantic similarity: words that are closer together in embedding space tend to have similar meaning. Formally, if two strings of text are represented by the vectors \mathbf{a} and \mathbf{b} , then their cosine similarity $\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}$ measures how semantically related they are—with 0 being completely orthogonal and 1 being identical.

Table 1 provides several examples in which the cosine similarity between text embeddings provides a better measure of match quality than lexical similarity (see the next section for details on how these cosine similarities are computed). Consider, for example, the problem of linking an organization’s full name with its acronym (first four rows). Lexical measures of string distance will struggle with this sort of record linkage task, since an organization’s acronym may be lexically more similar to the acronym of another organization than it is to its own full name! By contrast, embedding vectors can encode the fact that AARP stands for American Association of Retired Persons, by representing those strings as vectors close to one another in space—this is how language models based on such embeddings (e.g., ChatGPT) “know” the relationship between those two concepts. In each of the examples in Table 1, the cosine similarity between text embeddings chooses the correct match, while lexical measures of string similarity do not. Consequently, a record linkage proce-

cedure that incorporates this measure of similarity may significantly outperform procedures that rely exclusively on lexical similarity.

Table 1: Examples where lexical similarity is a misleading measure of match quality. The first row of each pair is the true match, and the best match according to four string similarity measures is in bold. In each case, lexical measures select the wrong match, while the cosine similarity between pretrained text embeddings selects the correct match.

String 1	String 2	Levenshtein	Jaro-Winkler	Jaccard	Embedding
AARP	American Association of Retired Persons	0.103	0.517	0.188	0.837
AARP	AAA	0.500	0.722	0.333	0.555
USPS	US Post Office	0.214	0.655	0.250	0.814
USPS	UPS	0.750	0.806	1.000	0.753
Mike Kelly	George Joseph "Mike" Kelly, Jr.	0.323	0.354	0.421	0.827
Mike Kelly	Mark Edward Kelly	0.471	0.757	0.538	0.616
Kit Bond	Christopher Samuel Bond	0.304	0.475	0.368	0.605
Kit Bond	Katie Britt	0.364	0.627	0.455	0.445

This is not the first paper to propose using text embeddings for record linkage. Indeed, there is by now an extensive literature applying transformer models to what computer scientists call *entity resolution*—determining whether two or more entries in a large dataset refer to the same entity (Zhou et al., 2021; Tang et al., 2022). These models have had significant practical applications in areas like e-commerce, where merging product records across multiple websites is a challenging large-scale problem. These approaches have been adapted to social science applications as well, most notably in the work of Arora and Dell (2023). What distinguishes the current paper from previous work is that it incorporates embedding similarity into a probabilistic record linkage procedure. Such procedures are preferable in social science for two main reasons: they do not rely on arbitrary thresholds to determine whether two records constitute a match, and they allow post-merge analyses to account for uncertainty introduced during record linkage (Enamorado, Fifield and Imai, 2019). For applications where there may be multiple correct matches for each observation, a method that can estimate match probabilities will provide a principled approach for determining which records to merge, and how strongly to weight each observation in a subsequent analysis.

In this paper, I propose a probabilistic record linkage procedure that incorporates pretrained text embeddings into an active learning algorithm (Enamorado, 2018; Bosley et al., 2022). The approach, which I call **fuzzylink**, is a variant of Adaptive Fuzzy String Matching (Kaufman and Klevs, 2022), an iterative process of fitting a model, labeling uncertain matches, refining the model,

and repeating until the model converges. The labeling step is performed by zero-shot prompts to a language model, which reduces time and expense compared to hand-labeling (Ornstein, Blasingame and Truscott, 2025). Across a series of political science applications, I show that this approach significantly improves both precision and recall over existing approaches, and can even perform some tasks—like multilingual record linkage—that would be impossible using lexical similarity measures alone. In this paper I focus on applications with a single fuzzy matching variable (and potentially multiple exact “blocking” variables), and conclude by discussing how one might extend the procedure to multiple fuzzy matching variables.

2 The Algorithm

Consider the problem of merging two datasets \mathcal{A} and \mathcal{B} , with sample sizes $n_{\mathcal{A}}$ and $n_{\mathcal{B}}$ respectively. Let X be a matrix of predictors measuring the similarity between each record pair in the set $\mathcal{A} \times \mathcal{B}$. The goal of a probabilistic record linkage procedure is to estimate a model $f(X)$ that maps X onto a match probability for each record pair.

The workhorse model for this class of problem was first formalized by Fellegi and Sunter (1969). The Fellegi-Sunter model is an *unsupervised* approach to record linkage, because it does not require the researcher to provide labeled data on the true matching status of record pairs. Instead, the model estimates match probabilities using unlabeled data, requiring the researcher to pre-specify a set of discrete thresholds for what level of similarity constitutes a match. One advantage of this approach is that it can incorporate similarity metrics from many different types of variables, including strings, numbers, and geographic coordinates. Another key advantage is its computational efficiency. The **fastLink** implementation of the Fellegi-Sunter model by Enamorado, Fifield and Imai (2019) can easily handle merging large-scale administrative datasets with hundreds of millions of observations.

However, an unsupervised approach can be an inappropriate choice for record linkage problems with a single fuzzy matching variable—like the applications described in this paper—because without overlapping information from multiple matching variables, the model’s accuracy will be quite sensitive to the researcher’s choice of similarity thresholds. In such cases, one will generally prefer a *supervised* approach, in which the model learns the mapping between string similarity and match

probability based on labeled data.

The main practical impediment to a supervised approach is that the total number of record pairs scales with $n_{\mathcal{A}} \times n_{\mathcal{B}}$, so it quickly becomes infeasible to label every record pair even in small-scale applications. The general solution to such problems is active learning (Enamorado, 2018; Bosley et al., 2022). Rather than exhaustively labeling every pair of records, an active learning approach identifies the most *informative* record pairs with which to train the model—that is to say, the record pairs for which the model is most uncertain. By iteratively fitting a model, selecting informative pairs to label, refitting the model, and repeating until the model converges, one can train a supervised learner using a relatively small number of observations.

The active learning algorithm described below performs a fuzzy “left join”, identifying every record in \mathcal{B} that matches at least one record in \mathcal{A} . It proceeds in six steps.

Step 1: Embedding. Select the string variable that identifies each record in \mathcal{A} and \mathcal{B} , and retrieve text embeddings for each unique string. In the analyses that follow, I use 256-dimensional pretrained embeddings from OpenAI.¹ Wherever possible, the strings representing records should *not* be pre-processed by stemming, converting to lowercase, or any other steps that one might take to reduce complexity in a bag-of-words representation (Grimmer and Stewart, 2013); performance will generally be improved if we embed text as it is most likely to appear in the training corpus (e.g. “Coca-Cola” instead of “cocacola”). The output from this step will be two matrices $\mathbf{M}_{\mathcal{A}}$ and $\mathbf{M}_{\mathcal{B}}$, with dimensions $n_{\mathcal{A}} \times 256$ and $n_{\mathcal{B}} \times 256$ respectively. Each row of these matrices is an embedding vector.²

Step 2: Compute Similarity Metrics. For each pair of records in the set $\mathcal{A} \times \mathcal{B}$, compute the cosine similarity between their embedding vectors. If the embeddings are normalized to length 1, a matrix of cosine similarities can be efficiently computed by taking the product $\mathbf{M}_{\mathcal{A}}(\mathbf{M}_{\mathcal{B}})'$. If there are any variables that must match exactly to link a record from \mathcal{A} to \mathcal{B} (“blocking variables”), perform this step only for pairs of records with exact matches on these variables. Since the computational

¹The most up-to-date embedding model offered by OpenAI as of February 2025 returns 3,072-dimensional embeddings, but one can reduce the dimensionality through “Matryoshka Representation Learning” (Kusupati et al., 2024), dramatically improving computation speed at little cost to accuracy. The most recent training data for these embedding models is September 2021, meaning the approach will underperform if successfully linking records requires knowledge of events that have occurred since that date.

²An alternative approach to computing embedding similarity—called cross-encoders (Lin, 2025)—is to pass string pairs directly to a transformer model, outputting a similarity score. Although such an approach could improve accuracy, it would come at the cost of quadratic computational complexity during the embedding step, so I do not implement it here.

complexity of this step scales with $n_{\mathcal{A}} \times n_{\mathcal{B}}$, blocking can significantly improve efficiency and as practical matter should be used whenever possible.

Step 3: Label a Training Set. Select a subset of record pairs and assign each pair a binary label, 1 if the records are a true match and 0 otherwise. For this paper’s analyses, I begin with an initial training set of the 500 record pairs with the highest cosine similarity scores and generate labels using the following zero-shot prompt to OpenAI’s GPT-4o:³

Decide if the following two names refer to the same {record_type}.
 {additional_instructions} Think carefully.⁴ Respond with "Yes" or "No".

Name A: {A}

Name B: {B}

The placeholders {record_type} and {additional_instructions} will vary by application. The accuracy of LLM labels is often improved by including context-specific instructions or examples (Ornstein, Blasingame and Truscott, 2025), just as a researcher would include a detailed codebook if this step were conducted by human research assistants or crowd-coders.

Step 4: Fit Supervised Learner. Fit a probabilistic model to map these cosine similarities onto a match probability. In the analyses that follow, I fit a logistic regression, which has the advantage of being significantly faster at generating predictions for large datasets than other supervised learners. I include as predictors both embedding similarity and Jaro-Winkler similarity, to capture both semantic and lexical differences between records.⁵

Step 5: Label Informative Cases. Estimate match probabilities for all record pairs in the set $\mathcal{A} \times \mathcal{B}$ using the fitted model from Step 4. Select N_L record pairs to label using uncertainty sampling, where selection probability is determined by a Gaussian kernel centered on match probability of $\frac{1}{2}$ (Enamorado, 2018).⁶ Assign labels to these record pairs as in Step 3. Add the new labeled

³In the Supplementary Materials, I replicate the paper’s empirical applications using open-source language models for the embedding and labeling steps. The advantage of open-source models is that their results are fully reproducible, though this comes at the expense of poorer record linkage accuracy. I discuss this tradeoff more fully in Section 4.

⁴Bizarre as it may seem, prompts that include phrases like “Think carefully” often yield marginal gains in classification accuracy (Battle and Gollapudi, 2024).

⁵In Section A.1 of the Supplementary Materials, I vary the model specification in Step 4 and show that this choice yields the best-calibrated probability estimates.

⁶Formally, selection probability is based on estimated log-odds. I use the kernel $\mathcal{N}(0, 0.2)$, which has 95% of its mass between 40% and 60% match probability.

observations to the training set and refit the model as in Step 4. Repeat these steps until a stopping criterion is met. In this paper’s analyses, I label $N_L = 100$ record pairs per active learning iteration, and stop when none of the estimated probabilities $f(X)$ change by more than 0.01 between iterations.

Step 6: Link Datasets. If the number of labeled record pairs is fewer than a researcher-specified budget N_{\max} , continue labeling record pairs from \mathcal{A} without an identified match in \mathcal{B} . This will generally improve recall by identifying true matches with low estimated match probabilities. Return all record pairs and their estimated match probability. Optionally, only return record pairs with an estimated match probability greater than π , where π is selected to maximize expected F_1 score.⁷

3 Applications

In this section, I describe four applications of the method, testing its performance across a variety of record linkage tasks common in political science. The first application merges the names of over 9,000 candidates for public office with voter file records from tens of millions of registered voters in California. The second application replicates an analysis merging misspelled names of US cities with a dataset of place names from the US Census Bureau. The third application merges the names of interest groups with ideology scores estimated from campaign contributions. And the final application explores how well the method can perform record linkage across multiple languages, merging the names of political parties from 32 countries in 30 different languages.

For each application, I evaluate performance by computing both precision and recall, where precision measures the fraction of identified matches that are correct, and recall measures the fraction of correct matches that are identified.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

⁷The F_1 score is the harmonic mean of precision and recall, as defined in Section 3.3. This step will typically remove a large number of record pairs to which the model assigns a very low match probability, significantly reducing the difficulty of post-merge manual validation (Section 4).

Any method that performs well on both of these metrics is likely to be particularly useful for researchers. Higher precision increases the quality of matches, reducing measurement error in subsequent empirical analyses. Higher recall reduces the amount of missing data in the linked dataset, increasing the statistical power of downstream analyses (Kaufman and Klevs, 2022)—as well as reducing bias whenever that missingness is non-random.

3.1 Linking Candidates to Voter File Records

Every year, hundreds of thousands of candidates are elected to local public office throughout the United States. Collecting data on these elections can be a painstaking process (Sumner, Farris and Holman, 2020; Einstein, Ornstein and Palmer, 2022; de Benedictis-Kessner et al., 2023), because unlike candidates for state and federal office, there is often very little information recorded about local candidates except their names. In this application, I merge the names of every candidate for mayor and city council in the state of California since 2016 with their corresponding records in the L2 voter file. There are a total of 9,025 unique candidate names, and roughly 22 million registered voters in the California voter file.⁸ I merge these two datasets using full name as the fuzzy matching string and exact blocking on last name and city of residence. To make validation feasible, the author and a research assistant hand-coded matches from three counties—Alameda, Kern, and Ventura—to estimate precision and recall.

Of the 840 candidates that ran for office in these three counties, **fuzzylink** identified 770 potential matches in the voter file. 154 of these were exact matches, and the research team determined that 584 of the remaining fuzzy matches were valid, for an estimated precision of 95.8%. In addition, the research team was able to locate 32 matches in the L2 voter file that **fuzzylink** failed to identify,⁹ for an estimated recall rate of 95.8%. By comparison, the **fastLink** approach (Enamorado, Fifield and Imai, 2019)—which links records based on predetermined cutoffs in Jaro-Winkler scores¹⁰—identifies only 521 potential matches, with an estimated precision of 93.3% and recall of 63.1%. The dramatically improved recall is largely due to **fuzzylink** successfully linking a variety of nicknames from the candidate list with legal names in the voter file (e.g., “Vinnie” with “Vinton”, “Chuck” with “Charles”, “Libby” with “Elizabeth”, “Trish” with “Patricia”, “Mel” with “Carmelita”,

⁸The California Election Data Archive (CEDA) is available at <http://www.csus.edu/isr/projects/ceda.html>.

⁹This search was conducted with the aid of local newspaper articles, campaign websites, and obituaries.

¹⁰I use the package’s default thresholds of 0.88 for a partial match and 0.94 for a full match.

“Sri” with “Sricharana”, “Teddy” with “Theadora”). There are also a number of cases where candidates go by their middle name (e.g., “Gregory Tod Abbott” listed as “Tod Abbott” on the ballot) and are correctly paired by the LLM prompt.

It is worth noting, in light of ongoing debates over algorithmic bias in language models (Abid, Farooqi and Zou, 2021; Grossmann et al., 2023), that a disproportionate share of false positive matches (26 out of 32) are Asian, Hispanic, or African American names. As with any record linkage procedure, researchers should take the time to carefully examine a subset of the merged dataset and ensure that the method is performing as expected. Fortunately, the estimated match probabilities are well-calibrated (see Appendix Section A.1) and can serve as a useful guide during validation: the false positives had a median match probability of just 22%, compared to 58% for the true positives. One could eliminate over half of the false positives in the merged dataset by manually validating only the 188 least-probable matches.

3.2 Linking Misspelled City Names To US Census Bureau Records

Next, I replicate a record linkage task from Kaufman and Klevs (2022), which allows for a direct comparison between the two approaches. There are four key differences between the **fuzzylink** algorithm and the AFSM algorithm proposed by Kaufman and Klevs (2022): (1) the inclusion of embedding similarity as a predictor of match quality, (2) the use of automated labeling by LLMs instead of human coders, (3) a logistic regression classifier instead of random forest, and (4) selecting record pairs to label through uncertainty sampling. See Appendix A for a detailed ablation analysis, exploring the effect of each of these choices.

The first dataset contains information on 661,218 loan recipients from the 2021 Paycheck Protection Program (PPP) implemented by the US federal government in the wake of the COVID-19 pandemic. This dataset contains each recipient’s address, but the city name provided is rarely an exact match with place names as listed by the US Census Bureau. If a researcher wanted to determine which US municipalities were receiving funds through this program, it would require linking 7,118 misspelled city names in the PPP records with a place-level dataset of 28,889 incorporated towns and cities maintained by the Census, blocking by state.

This is another application where methods that rely on lexical similarity alone can fall short, because many pairs of cities have quite similar names, potentially yielding a large number of false

positive matches. Replicating the AFSM procedure as described in Kaufman and Klevs (2022) yields a set of 1,075 city pairs for which the model assigns a match probability greater than 90%. Of these record pairs, only 705 were confirmed as true matches by the research team, for an estimated precision of 66%. Some examples of the 370 false positive matches include Wingdale NY \rightarrow Walden NY, Deerpark TX \rightarrow Parker TX, Maple TX \rightarrow Palmer TX, Lamont MI \rightarrow Almont MI, Delair NJ \rightarrow Garfield NJ, and Malone WI \rightarrow Montreal WI.

Adding embedding similarity as a predictor in the AFSM algorithm significantly improves performance, increasing precision to 84% and the number of matches correctly identified to 1,049 (see Table 2). But despite this improvement, the approach still struggles with more challenging cases, yielding a number of false positive matches like Saint Augustine FL \rightarrow Saint Augustine Beach FL, Preston CT \rightarrow New Preston CT, and Swanzey NH \rightarrow West Swanzey NH.

Table 2: Performance metrics for city name merge across three algorithms.

Algorithm	True Matches Identified	Precision
AFSM	705	65.6%
AFSM with Embeddings	1,049	84.1%
fuzzylink	2,451	98.0%

The **fuzzylink** algorithm dramatically improves over both these approaches, more than doubling the number of true matches recalled with near-perfect precision. Key to this performance is the combination of uncertainty sampling and accurate LLM labeling, so that challenging cases like the ones mentioned above are identified and correctly labeled by the LLM during the active learning loop. Not only is precision improved by removing these false positives, but recall is improved by identifying true matches with low lexical similarity. These include OKC \rightarrow Oklahoma City, Olympic Valley WA \rightarrow Squaw Valley WA, and USAF Academy CO \rightarrow Air Force Academy CO. The algorithm also correctly pairs cases where the PPP loan recipient listed a neighborhood rather than a city in their address, like Astoria NY \rightarrow Queens NY, Newbury Park CA \rightarrow Thousand Oaks CA, and Port Bolivar TX \rightarrow Bolivar Peninsula TX.

3.3 Linking Amicus Cosigners to Campaign Donations

For the paper’s third application, I replicate the record linkage from Abi-Hassan et al. (2023), who estimate the ideology of interest groups by merging the names of organizations that cosigned

Supreme Court amicus curiae briefs (Box-Steffensmeier, Christenson and Hitt, 2013) with ideal point estimates (DIME scores) from campaign donations (Bonica, 2014). There are 15,376 organizations in their dataset and 2.9 million organizations with recorded campaign donations in the DIME dataset. The scale of these datasets poses a significant practical challenge for computation and validation—without any blocking variables, linking the full versions of both datasets requires computing approximately 38 billion pairwise similarity scores. To make manual validation feasible, I focus here on the 1,388 organizations that cosigned amicus briefs in the year 2012, and to reduce computational complexity, I also restrict the DIME dataset to organizations with at least eight distinct campaign contributions. This is both a practical and principled choice, since “donating to eight or more distinct recipients is [typically] sufficient to recover a reliable ideal point estimate” (Bonica, 2023).

Through a combination of exact matching and fuzzy string matching, Abi-Hassan et al. (2023) were able to locate DIME scores for 376 of these 1,388 organizations, approximately 27% of the total. By comparison, despite restricting its search to only 8% of the DIME dataset, **fuzzylink** is able to locate DIME scores for 444 unique organizations. As in the first application, this dramatically improved recall is largely the result of correctly identifying alternative names for the same organization (e.g., “Utah Association for Justice” and the “Utah Trial Lawyers Association”, “California Forestry Association’ and “CA Forestry Assoc PAC”, “Ojibwe” and “Chippewa” tribes) and even former names of the same organization (e.g., “The Association of Magazine Media” formerly “Magazine Publishers Association”, “California Construction Trucking Association” formerly “California Dump Truck Owners Association”, “United States Telecom Association” formerly “United States Telephone Assn”, “PacifiCorp” formerly “Pacific Power & Light”, “RELX” formerly “Reed Elsevier”). This improved recall does not appear to come at the expense of precision: the research team identified only 7 false positives out of 1,058 proposed matches, for an estimated precision of 99.3%. Note that this estimate considers chapters or subsidiaries of larger organizations to be true matches (for example, linking “NAIOP” and “NAIOP New Jersey Chapter”), under the assumption that one can use campaign donations of local chapters to make inferences about the parent organization’s ideology. If one were unwilling to make such an assumption, those matches could easily be filtered out post-merge, or one could modify the LLM prompt in Step 3 to ignore such matches.

3.4 Linking Political Party Names Across Multiple Languages

For record linkage problems involving multiple languages, lexical similarity measures tend to be a poor guide to match quality. The strings “LDP” and “Jiyū Minshutō”, for example, share no lexical features at all, but both refer to the same Japanese political party. Pretrained text embeddings, by comparison, can naturally accommodate this sort of problem by representing text from multiple languages in the same embedding space. This makes transformer models particularly adept at machine translation tasks (Vaswani et al., 2017). In this application, I demonstrate that the approach proposed here can successfully link the names of political parties across 30 languages—though performance is better for some languages than for others.

To test the method, I take the ParlGov dataset of parliamentary elections since 1900 (Döring and Manow, 2018), splitting it into two datasets as illustrated in Table 3. The first dataset contains each party’s name in its native language, the election year, and the number of seats the party won in parliament that year. The second dataset contains the English translation of the party’s name along with its estimated left-right ideology on a scale from 0 (leftmost) to 10 (rightmost). I include all parties from non-English speaking countries that won seats in parliament, for a total of 4,972 observations across 32 countries and 663 elections. Because text embeddings may be closer in space for some language pairs than others¹¹, I perform this record linkage separately for each country, blocking on election date.

The resulting dataset correctly matches 4,855 name pairs out of 4,972—a recall rate of 97.6%. There are, however, a large number of false positive matches (453 in total), for an overall precision of 91.5%. As expected, the method’s accuracy varies somewhat by language: precision and recall are lower for countries like Israel (68.9% precision, 87.0% recall) and Japan (79.1% precision and 94.6% recall) than for Italy (98.3% precision, 99.6% recall) or Portugal (100% precision, 100% recall). See Appendix Table A4 a complete list of these evaluation metrics by country.

In addition to computing these accuracy metrics, one can evaluate whether the record linkage procedure allows us to recover downstream quantities of interest. Figure 1 plots the seat-share weighted ideology of every parliament in the ParlGov dataset (lines) along with each parliament’s

¹¹For example, as measured by cosine similarity, the phrase “Social Democratic Party” is much closer to the Portuguese “Partido Social Democrata” (0.80) than it is to the Icelandic Social Democratic Party “Alþýðuflokkurinn” (0.44). However, “Alþýðuflokkurinn” is closest to “Social Democratic Party” *relative* to other Icelandic parties, so the probabilistic model will perform best if we avoid pooling embedding distances across language pairs.

Table 3: Multilingual Record Linkage Application: Splitting the ParlGov data into two sets with 4,972 observations each. The first dataset contains each party’s name in the country’s native language, and the second dataset contains the English name of each political party.

Native Party Names:

country_name	election_date	party_name	seats
Austria	1919-02-16	Sozialdemokratische Partei Österreichs	72
Austria	1919-02-16	Österreichische Volkspartei	69
Austria	1919-02-16	Deutschnationale	8
Austria	1919-02-16	Deutsche Freiheits und Ordnungspartei	5
.	.	.	.
.	.	.	.
Turkey	2023-05-14	Zafer Partisi	0

English Party Names:

country_name	election_date	party_name	left_right
Austria	1919-02-16	Social Democratic Party of Austria	3.7293
Austria	1919-02-16	Austrian People’s Party	6.4733
Austria	1919-02-16	German-Nationals	7.4000
Austria	1919-02-16	German Freedom and Order Party	8.8000
.	.	.	.
.	.	.	.
Turkey	2023-05-14	Victory Party	8.8000

estimated ideology following the record linkage (points).¹² The correlation between the estimates and their true values is 0.964, and the estimates are perfectly correlated with the truth in most countries. Only a few country-years stand out as severely mis-estimated. In Switzerland, the model incorrectly links the FDP (“Freisinnig-Demokratische Partei der Schweiz”) with both the Liberal Party of Switzerland and the Radical Democratic Party. These two parties merged in 2009, but they were separate parties throughout the prior century, which biases our estimates rightwards for much of the 20th century. In Israel the model incorrectly links the Labor Party (“HaAvoda”) with the right-wing Likud, and in Turkey the model fails to identify a match for the Social Democratic Populist Party. In both cases these errors bias the estimates rightward.

In practice, errors like these can be easily corrected by conducting a post-merge manual validation, focusing on records in \mathcal{A} that did not match to a single unique record in \mathcal{B} . In this case, it would require manually checking only 128 proposed matches, roughly 2% of the total.

¹²For each party in \mathcal{A} , estimated ideology is computed as the average ideology of its matches in \mathcal{B} , weighted by match probability.

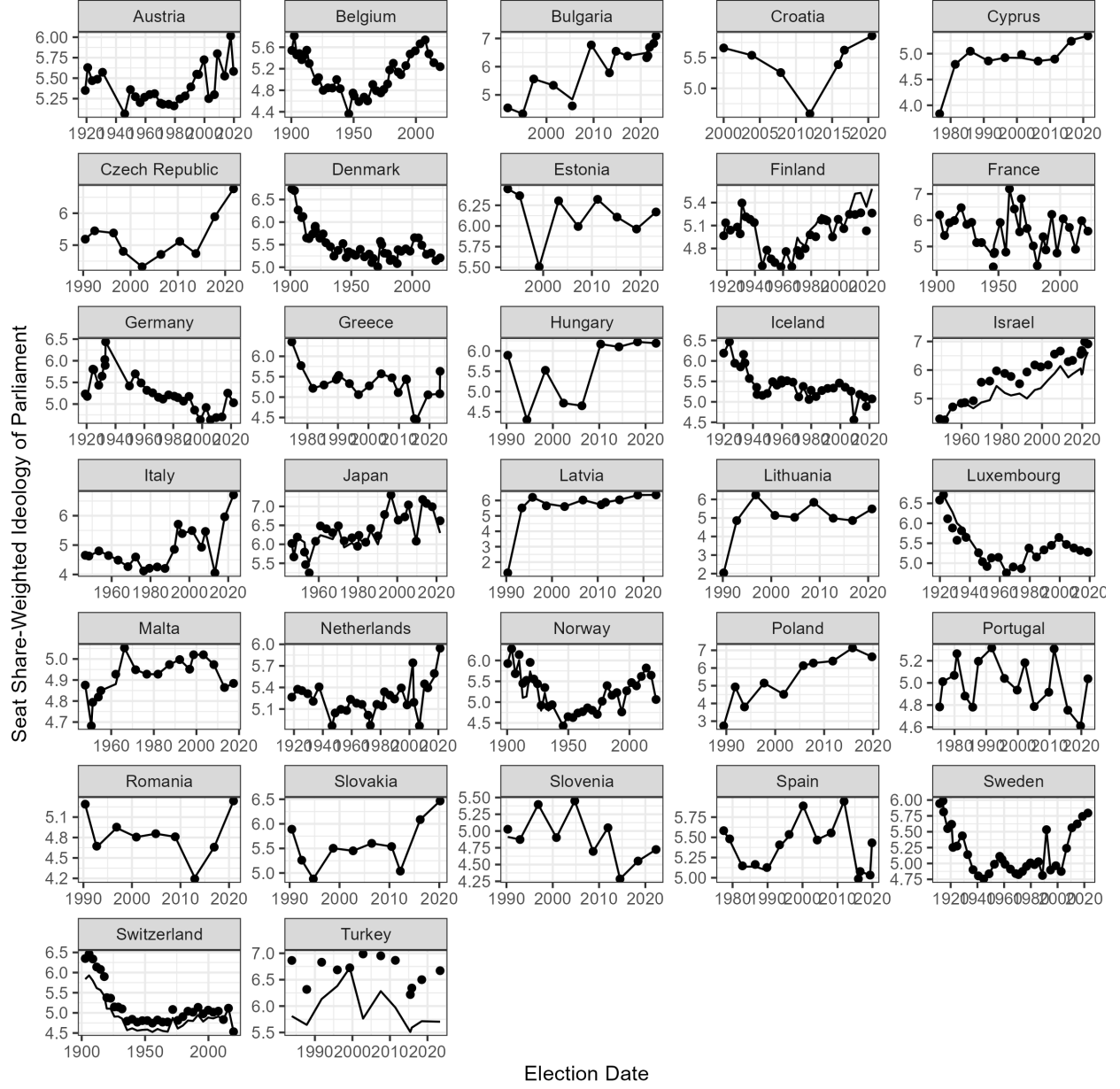


Figure 1: Estimated seat-weighted parliamentary ideology following merge (points) plotted over true values (lines).

4 Discussion

The approach I propose here has significant advantages over methods that rely on lexical similarity measures alone. Social scientists often encounter record linkage problems where matching entities may be lexically dissimilar from one another, whether it’s due to alternative names, acronyms, or even different languages. Under such conditions, the **fuzzylink** procedure can significantly improve both precision and recall. And it does so without requiring significant expenditure in time or money. None of the applications described in the previous section took longer than a few hours to execute on a personal computer or cost more than \$10 in API fees (see Appendix B for a more thorough cost breakdown).

Despite these advantages, there are several situations in which the proposed approach is likely to fall short or prove unnecessary. For example, when merging large-scale administrative datasets with tens or hundreds of millions of records, researchers are likely to prefer the added efficiency of an unsupervised approach like **fastLink** (Enamorado, Fifield and Imai, 2019). Particularly when such datasets contain a large number of identifying fields, the marginal gains in accuracy from a supervised approach like **fuzzylink** are unlikely to be worth the loss of efficiency. The proposed approach is also likely to fail when pretrained language models do not encode the relevant world knowledge necessary to link two records. The embeddings used in this paper, for example, are trained only on data collected before September 2021, and will therefore struggle to perform any record linkage task that requires knowledge of events after that date. Post-merge clerical review is essential to ensure the accuracy of the LLM labels, replacing them with human labels in cases where they perform poorly. Finally, researchers will find this approach unnecessary in applications where discrepancies between records are due solely to typos or misspellings, in which case embedding similarity offers little predictive advantage over lexical similarity alone.

I have focused in this paper on applications where there is a single fuzzy string matching variable, but the sorts of record linkage problems faced by social scientists often include many such variables. Fortunately, the method can be extended in a number of ways. One approach would be to re-express multiple fuzzy variables as a single string, which can then be represented as an embedding. For example, a record with {name} and {address} fields might be represented by the string “My name is {name} and I live at {address}.” Another approach would be to estimate a match probability

separately for each variable as I have done here, and then use those match probabilities as inputs in the Fellegi-Sunter model (Enamorado, Fifield and Imai, 2019). Further research is needed to determine which approach yields better results.

Another limitation of the method as presented is its reliance on proprietary language models. Because these models are closed-source and operated by for-profit entities, they can be deprecated or modified at any time without the consent of their users. Consequently, the results that **fuzzylink** produces—including those presented in this paper—are not fully reproducible. Though a researcher could replicate the steps I used to generate the results, within a few years it will be impossible to reproduce them exactly. For this reason, many scholars in our discipline have urged using open-source language models wherever possible (Spirling, 2023).

Unfortunately, as of writing, it is difficult to see how the method presented here could be undertaken using open-source language models. Frankly, the level of accuracy I demonstrate here would not have been possible even using the previous generation of *proprietary* language models. In the Supplementary Materials, I attempt to replicate the paper’s empirical applications using one of the highest performing open-source language models currently available (Mistral 8x22B), as well as the previous generation of language models released by OpenAI as of early 2023 (GPT-3.5). These variants significantly underperform the results reported in the previous section, particularly for the organization matching and multilingual record linkage applications. Given the rapid development of open-source language models, it is likely that there will be an acceptable open-source solution in the coming years, but until that time the accuracy gains from proprietary models outweigh their drawbacks.

When a research method falls short of full computational reproducibility, one must insist that it meet standards of *replicability* (procedures are transparently documented so that other scholars can independently replicate them) and *reliability* (repeated application of the procedure yields similar, if not identical, outcomes). Indeed, these are the standards that our discipline applies to other non-reproducible research methods, like those that rely on human research assistants or crowd-coders. The **fuzzylink** software package¹³ was developed to help researchers implement the method proposed here in a straightforward and replicable manner, and I hope that it will enable

¹³Implemented in the R programming language, the package is available at <https://github.com/joeornstein/fuzzylink>.

much useful social science research in the coming years.

References

- Abi-Hassan, Sahar, Janet M. Box-Steffensmeier, Dino P. Christenson, Aaron R. Kaufman and Brian Libgober. 2023. “The Ideologies of Organized Interests and Amicus Curiae Briefs: Large-Scale, Social Network Imputation of Ideal Points.” *Political Analysis* 31(3):396–413.
- Abid, Abubakar, Maheen Farooqi and James Zou. 2021. “Large Language Models Associate Muslims with Violence.” *Nature Machine Intelligence* 3(6):461–463.
- Arora, Abhishek and Melissa Dell. 2023. “LinkTransformer: A Unified Package for Record Linkage with Transformer Language Models.”.
- Battle, Rick and Teja Gollapudi. 2024. “The Unreasonable Effectiveness of Eccentric Automatic Prompts.”.
- Bonica, Adam. 2014. “Mapping the Ideological Marketplace.” *American Journal of Political Science* 58(2):367–386.
- Bonica, Adam. 2023. “Database on Ideology, Money in Politics, and Elections.”.
- Bosley, Mitchell, Saki Kuzushima, Ted Enamorado and Yuki Shiraito. 2022. “Improving Probabilistic Models in Text Classification via Active Learning.”.
- Box-Steffensmeier, Janet M., Dino P. Christenson and Matthew P. Hitt. 2013. “Quality Over Quantity: Amici Influence and Judicial Decision Making.” *American Political Science Review* 107(3):446–460.
- de Benedictis-Kessner, Justin, Diana Da In Lee, Yamil R. Velez and Christopher Warshaw. 2023. “American Local Government Elections Database.” *Scientific Data* 10(1):912.
- Döring, Holger and Philip Manow. 2018. “Parliaments and Governments Database (ParlGov): Information on Parties, Elections and Cabinets in Modern Democracies.” *ParlGov* .
- Einstein, Katherine Levine, Joseph T. Ornstein and Maxwell Palmer. 2022. “Who Represents the Renters?” *Housing Policy Debate* pp. 1–15.

- Enamorado, Ted. 2018. “Active Learning for Probabilistic Record Linkage.” *SSRN Electronic Journal*.
- Enamorado, Ted, Benjamin Fifield and Kosuke Imai. 2019. “Using a Probabilistic Model to Assist Merging of Large-Scale Administrative Records.” *American Political Science Review* 113(2):353–371.
- Fellegi, Ivan P. and Alan B. Sunter. 1969. “A Theory for Record Linkage.” *Journal of the American Statistical Association* 64(328):1183–1210.
- Grimmer, Justin and Brandon M. Stewart. 2013. “Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts.” *Political Analysis* 21(3):267–297.
- Grossmann, Igor, Matthew Feinberg, Dawn C. Parker, Nicholas A. Christakis, Philip E. Tetlock and William A. Cunningham. 2023. “AI and the Transformation of Social Science Research.” *Science* 380(6650):1108–1109.
- Jaro, Matthew A. 1989. “Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida.” *Journal of the American Statistical Association* 84(406):414–420.
- Kaufman, Aaron R. and Aja Klevs. 2022. “Adaptive Fuzzy String Matching: How to Merge Datasets with Only One (Messy) Identifying Field.” *Political Analysis* 30(4):590–596.
- Kusupati, Aditya, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain and Ali Farhadi. 2024. “Matryoshka Representation Learning.”
- Lin, Gechun. 2025. “Using Cross-Encoders to Measure the Similarity of Short Texts in Political Science.”
- Ornstein, Joseph T., Elise N. Blasingame and Jake S. Truscott. 2025. “How to Train Your Stochastic Parrot: Large Language Models for Political Texts.” *Political Science Research and Methods* pp. 1–18.
- Rodriguez, Pedro L. and Arthur Spirling. 2022. “Word Embeddings: What Works, What Doesn’t, and How to Tell the Difference for Applied Research.” *The Journal of Politics* 84(1):101–115.

- Spirling, Arthur. 2023. “Why Open-Source Generative AI Models Are an Ethical Way Forward for Science.” *Nature* 616(7957):413–413.
- Sumner, Jane Lawrence, Emily M. Farris and Mirya R. Holman. 2020. “Crowdsourcing Reliable Local Data.” *Political Analysis* 28(2):244–262.
- Tang, Jiawei, Yifei Zuo, Lei Cao and Samuel Madden. 2022. “Generic Entity Resolution Models.” *NeurIPS* .
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. 2017. “Attention Is All You Need.” *arXiv:1706.03762 [cs]* .
- Zhou, Huchen, Wenfeng Huang, Mohan Li and Yulin Lai. 2021. “Relation-Aware Entity Matching Using Sentence-BERT.” *Computers, Materials & Continua* 71(1):1581–1595.

A Ablation Study

The algorithm proposed in the paper differs from existing methods on several dimensions, and in this appendix I show how altering several of those choices affects performance. I focus here on five dimensions: (1) the similarity metrics used as predictors, (2) supervised vs. unsupervised learning, (3) active vs. passive labeling of record pairs, (4) the choice of supervised classifier, and (5) the choice of language model for embeddings and labels.

A.1 Similarity Measures

The proposed algorithm uses both embedding similarity and Jaro-Winkler similarity as predictors of match quality. In this analysis, I explore how using only embedding similarity or only Jaro-Winkler similarity affects performance.

In the main text, I argue that a key advantage of a probabilistic approach to record linkage is that the estimated match probabilities can be used in downstream statistical analyses as a measure of confidence in the match quality. But in order for match probabilities to be useful in this manner, they must be well calibrated—record pairs that `fuzzylink` assigns a 90% match probability should be true matches roughly 90% of the time. To assess calibration, I extract estimated match probabilities for every within-block pair of records from the first and third applications (linking candidates with voter file records and the multilingual record linkage application). I exclude record pairs that are exact matches, and compare the estimated match probabilities with the true rate of matching.

As Figures A1 and A2 illustrate, the estimated match probabilities are strongly correlated with hand-validated match rates, though they tend to be somewhat underconfident relative to perfect calibration (dashed line). In each figure, panel (a) plots the performance of the record linkage using the model specification as described in the main text—a logistic regression using a linear combination of embedding similarity and Jaro-Winkler similarity as predictors. Panel (b) uses only embedding similarity as a predictor, and panel (c) uses only Jaro-Winkler distance as a predictor.

Table A.1 reports two measures of calibration for each model variant: Brier Scores and Log Likelihoods. The Brier score is the mean squared error between a probability estimate and a binary outcome, ranging from 0 (the model perfectly predicts the outcome) to 1 (the model perfectly predicts the *opposite* outcome). Log likelihood measures how likely the observed outcomes are given

a set of probability estimates; higher values indicate a better fitting model. In both applications, the model variant that uses both embedding similarity and lexical similarity as predictors yields match probabilities that are better-calibrated than the other two model variants. Unsurprisingly, the model variant that only uses Jaro-Winkler similarity as a predictor performs significantly worse in the multilingual record linkage application (Figure A2).

Table A1: Estimated match probability calibration in two applications, varying model specification.

Application	Predictors	Brier Score	Log Likelihood
Voter File	Both	0.159	-560.09
Voter File	Embedding Similarity Only	0.158	-676.06
Voter File	Lexical Similarity Only	0.167	-762.64
Multilingual Record Linkage	Both	0.052	-8230.36
Multilingual Record Linkage	Embedding Similarity Only	0.055	-8641.85
Multilingual Record Linkage	Lexical Similarity Only	0.074	-11686.87

A.2 Supervision

The algorithm proposed here is a supervised approach to record linkage, in which a model is trained using labeled data to learn the mapping between similarity metrics and match probability. This is in contrast to an unsupervised approach, like the Fellegi-Sunter model, which relies on the researcher pre-specifying thresholds for what level of similarity defines a full or partial match. As described in the main text, supervised approaches sacrifice some computational efficiency for greater accuracy, particularly for applications that involve a single fuzzy identifying field.

To illustrate why a supervised approach can yield superior performance in the paper’s applications, Figure A3 plots the estimated mapping between embedding similarity and match quality across five applications: matching candidate names to the voter file, matching city names, and translating the names of political parties in Austria, Israel, and Spain. Note that it would be very difficult for a researcher to specify *ex ante* what level of embedding similarity constitutes a full or partial match, since the best values for thresholds change depending on application. For example, in the candidate name application, record pairs with embedding similarity greater than 0.892 have a greater than 75% probability of matching. But in the city names application, this value is 0.927. In the multilingual application, embedding similarity is systematically different for different language pairs, so that an appropriate threshold might be 0.815 for Austrian political parties and 0.999 for

Israeli political parties. These difficulties help explain why a supervised approach that learns the relationship between embedding similarity and match probability from data can yield significant performance improvements for these sorts of applications.

A.3 Active vs. Passive Labeling

As Enamorado (2018) notes, the added value of active learning for record linkage is efficiency—exhaustively labeling $\mathcal{A} \times \mathcal{B}$ record pairs is often infeasible even for modest sized datasets. Furthermore, the vast majority of record pairs are not *informative* observations for training a supervised learner. To illustrate this idea, consider Figure A4, which plots the distribution of embedding similarity and lexical similarity (Jaro-Winkler scores) for every within-block record pair in Application 1 (red are confirmed matches, and gray are non-matches). As the figure makes clear, there is a relatively small region in the upper right where the researcher’s labeling budget is most productively spent. Active learning is well-suited to identifying and labeling record pairs in that region.

For comparison, I conducted several ablation analyses in which record pairs from Application 1 were labeled “passively”: selecting a subset at random to label, fitting a logistic regression model to this training set, and otherwise linking the datasets as described in the main text. Table A.3 reports the estimated performance of these variants. One must label roughly ten times as many record pairs in this fashion before even approaching levels of precision and recall reported in the main text.

Table A2: Estimated performance for variants of the model with “passive” rather than active labeling.

Method	Number of Labels	Precision	Recall
fuzzylink	3,259	95.8%	95.8%
Passive Labeling (1%)	550	73.8%	74.8%
Passive Labeling (2.5%)	1,375	80.6%	71.2%
Passive Labeling (5%)	2,751	76.2%	75.2%
Passive Labeling (10%)	5,503	83.7%	72%
Passive Labeling (25%)	13,758	83.7%	79.8%
Passive Labeling (50%)	27,516	90.9%	85.6%
Passive Labeling (75%)	41,274	98%	89.5%

A.4 Classifier

The version of the algorithm presented in the main text uses a simple logistic regression model as its supervised learner, with embedding similarity and Jaro-Winkler similarity as predictors. Perhaps a more complex learner that incorporated additional measures of lexical similarity would do a better job estimating match probabilities. To test this, I replicated the first application (linking California candidates to the voter file) replacing the logistic regression model with a random forest as in Kaufman and Klevs (2022). In addition to embedding similarity, I include as predictors a suite of lexical similarity measures, including Jaccard distance, optimal string alignment, q-gram distance, cosine similarity, longest common substring distance, Jaro-Winkler similarity, and distance based on soundex encoding. See Kaufman and Klevs (2022) and the `stringdist` R package documentation for details on how these measures are computed.

There are two disadvantages to this approach that ultimately led me to prefer the logistic regression specification. First, random forest requires more labeled observations to fit a well-calibrated model; replicating the first application using random forest required 6,399 labeled record pairs, roughly twice the number of labels required in the main text. Second, the more complex model structure takes longer to compute predicted probabilities, which can meaningfully increase run time for larger-scale problems. For example, in Application 2 (merging city names), there are 5.5 million within-block pairs of city names; predicting match probabilities for these city pairs with random forest took 3.5 minutes on my personal computer—compared to 0.36 seconds with logistic regression. Multiplied over a few dozen active learning iterations, this represents a substantial increase in compute time for researchers.

This reduction in computational efficiency might be worthwhile if the random forest variant of the model yielded more accurate merges, but this does not seem to be the case. Replicating the first application yielded a merged dataset with an estimated precision of 92.8% (compared to 95.8% in the main text) and recall of 97.3% (compared to 95.8% in the main text). I include the random forest as a non-default option in the `fuzzylink` software package.

A.5 Alternate Language Models

A.5.1 Linking Candidates to Voter File Records

When linking these datasets using labels from GPT-3.5 instead of GPT-4, **fuzzylink** returns fewer matches, but with slightly better precision. Out of the 840 candidates that ran for office in the three California counties, the method identified 566 potential matches in the voter file. 154 of these were exact matches, and the research team determined that 411 of the non-exact matches were valid, for an estimated precision of 99.8%. However, this improved precision comes at the cost of recall. The research team was able to locate 139 matches in the L2 voter file that **fuzzylink** failed to identify, for an estimated recall rate of 80.3%.

When linking these datasets using labels and embeddings from the open-source Mistral 8x22B language model (released April 17, 2024), **fuzzylink** returns far fewer matches, but with improved precision. Out of the 840 candidates that ran for office in the three California counties, the method identified 518 potential matches in the voter file. 154 of these were exact matches, and the research team determined that 362 of the non-exact matches were valid, for an estimated precision of 99.6%. However, this improved precision comes at the cost of recall. The research team was able to locate 187 matches in the L2 voter file that the model failed to identify, for an estimated recall rate of 73.4%.

Linking these datasets using embeddings from the open-source Mistral 8x22B and labels from OpenAI’s GPT-4o, **fuzzylink** yields similar results as the analysis in the main text using both labels and embeddings from OpenAI. The algorithm returns 770 potential matches in the voter file. 154 of these were exact matches, and the research team determined that 584 of the non-exact matches were valid, for an estimated precision of 95.8%. The research team was able to locate 25 matches in the L2 voter file that the model failed to identify, for an estimated recall rate of 96.7%.

A.5.2 Linking Amicus Cosigners to Campaign Donors

When linking these two datasets using labels from GPT-3.5 instead of GPT-4, the method returns matches for a much larger number of organizations (695 instead of 444), but the precision of these matches is unacceptably low. In a random sample of 100, only 37% were deemed a valid match by the research team.

A.5.3 Linking Party Names Across Languages

When linking these two datasets using labels from GPT-3.5 instead of GPT-4, the method returns a tremendous number of false positive matches—6,284 in total. As a result, estimated recall is slightly higher than that reported in the paper (97.1%), but those true positives are swamped by false positives, for an estimated precision of 43.5%.

When linking these two datasets using labels and embeddings from the open-source Mistral 8x22B language model (released April 17, 2024), **fuzzylink** returns only 216 false positive matches, for an estimated precision of 95.2% (roughly as precise as the 95.8% reported in the main text). Recall, however, is somewhat lower, at 85.7%, and this reduced ability to identify matches yields significantly worse performance on the ideology estimation task—particularly in countries like Estonia, Finland, and Iceland—as illustrated in Figures A5 and A6.

B Costs

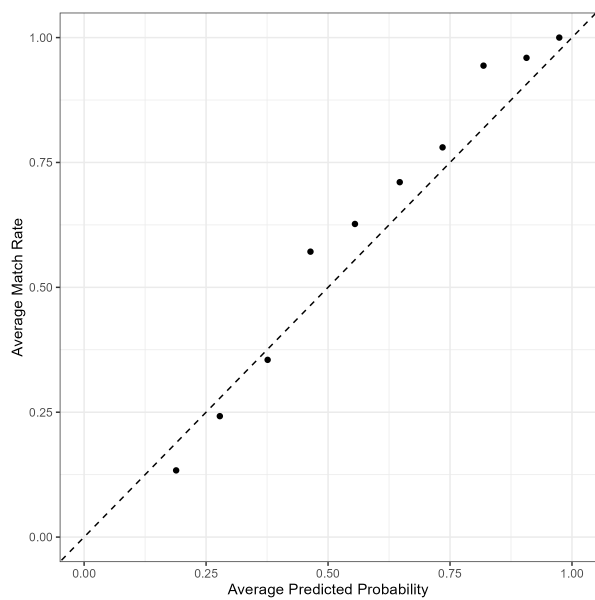
The following table reports the costs for each application in the paper, both in time and API fees. As described in the main text, blocking significantly improves efficiency. In the first application there are 773 blocks (determined by last name and place of residence) and the merge completes in less than 5 minutes. The third application, by comparison, has a similar number of records, but takes over 6.5 hours to complete because there are no blocking variables, and the algorithm must consider every possible pairwise comparison (approximately 260 million).

Table A3: Cost breakdown by application

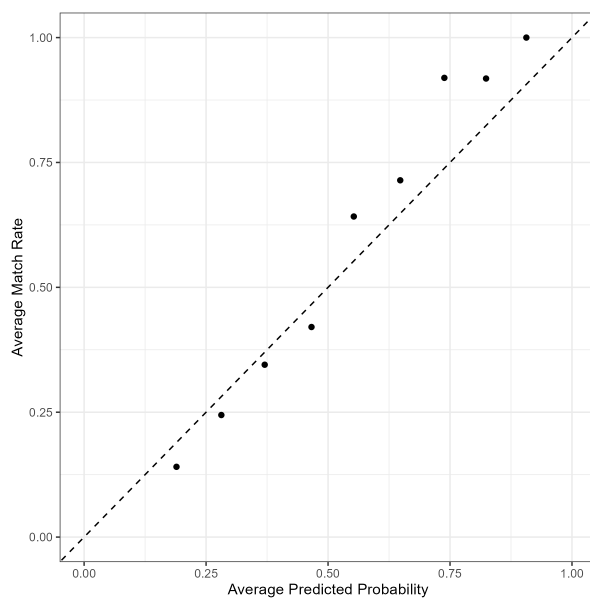
Application	$n_{\mathcal{A}}$	$n_{\mathcal{B}}$	Number of Blocks	Time	API Fees
1. L2 Voter File	1,019	1,868,345	773	4.2 minutes	\$1.83
2. PPP Cities	7,118	28,889	52	17 minutes	\$0.43
3. DIME scores	1,388	221,984	1	6.5 hours	\$10.05
4. ParlGov	4,972	4,972	32	14 minutes	\$1.05

C Additional Figures and Tables

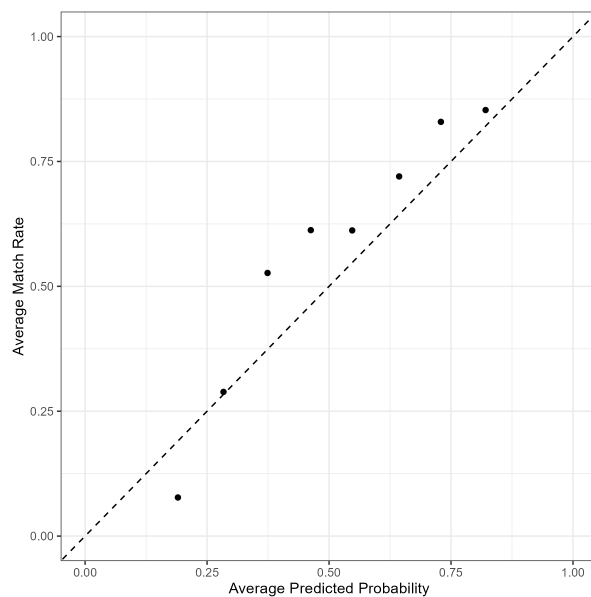
Figure A7 illustrates how embedding similarity can, in some applications, be a much better indicator of match quality than measures based on lexical similarity alone. The Receiver Operator Curves compare how well three similarity metrics predict the matching status of 4,000 hand-labeled pairs of organization names from Kaufman & Klevs (2022). The first (black) is lexical similarity alone (Jaccard similarity). The second is AFSM scores from Kaufman & Klevs (2022), which combine multiple lexical similarity metrics (red). The third is cosine similarity from pretrained text embeddings (blue). The Area Under the Curve (AUC) is a measure from 0 to 1 capturing how well the measure balances identifying true positives and eliminating false positives.



(a)



(b)



(c)

Figure A1: Binned calibration plot for voter file record linkage application, using three different sets of predictors: (a) both embedding similarity and Jaro-Winkler similarity, (b) embedding similarity only, and (c) Jaro-Winkler similarity only.

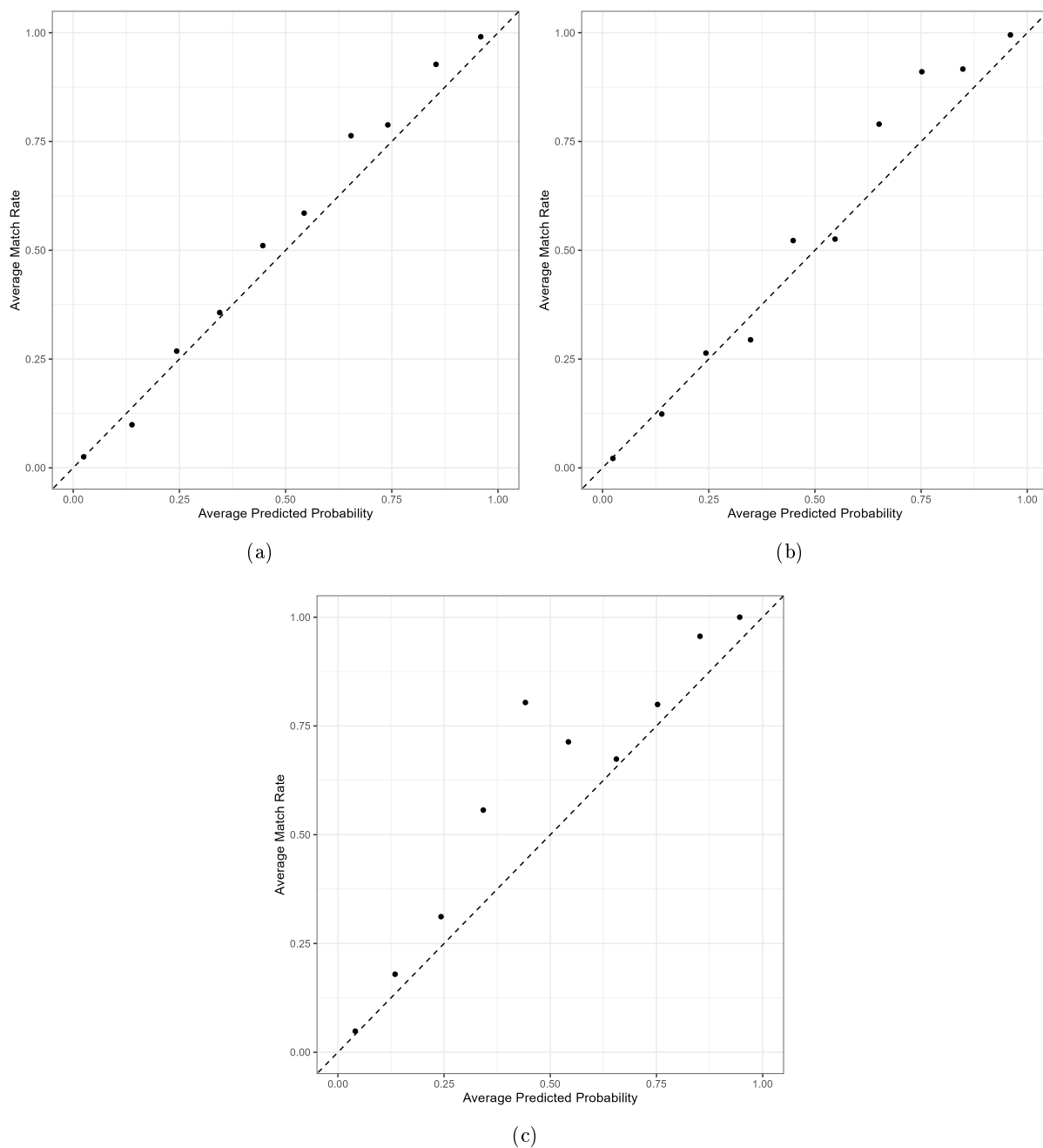


Figure A2: Binned calibration plot for multilingual record linkage application, using three different sets of predictors: (a) both embedding similarity and Jaro-Winkler similarity, (b) embedding similarity only, and (c) Jaro-Winkler similarity only.

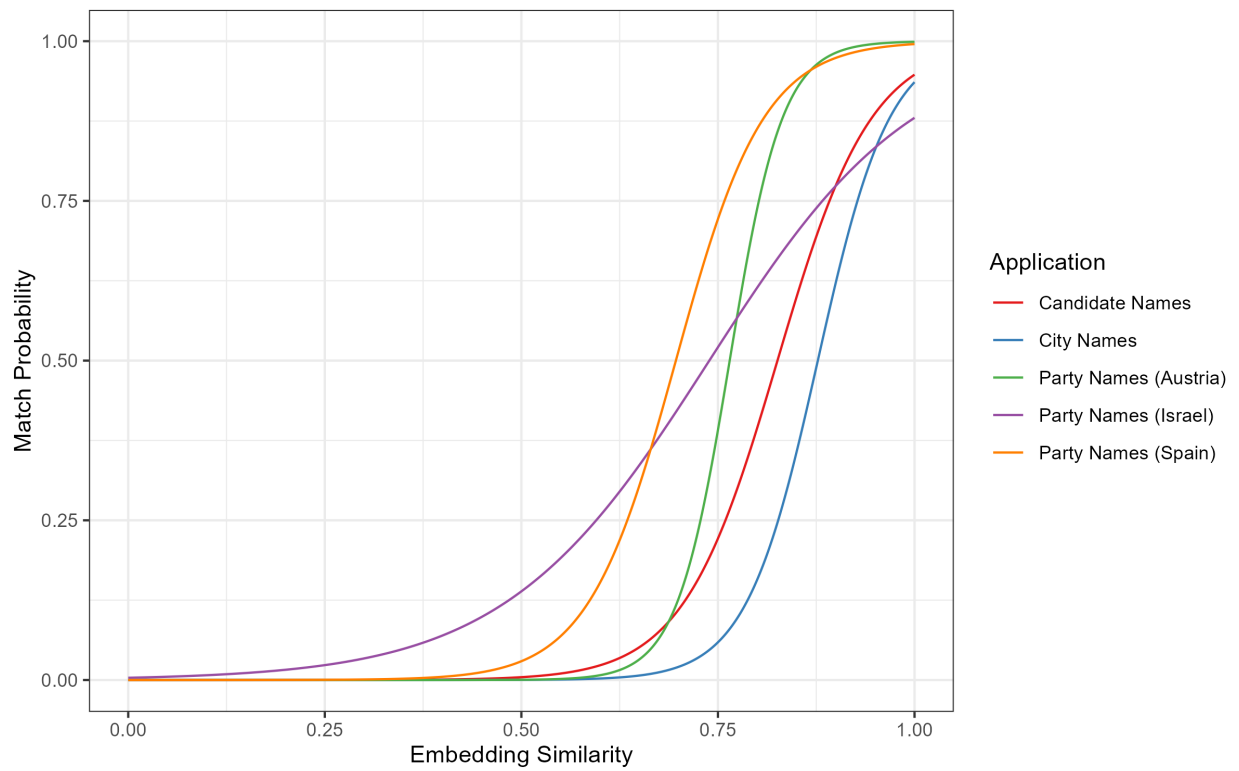


Figure A3: Learned mappings from embedding similarity to match probability in five applications

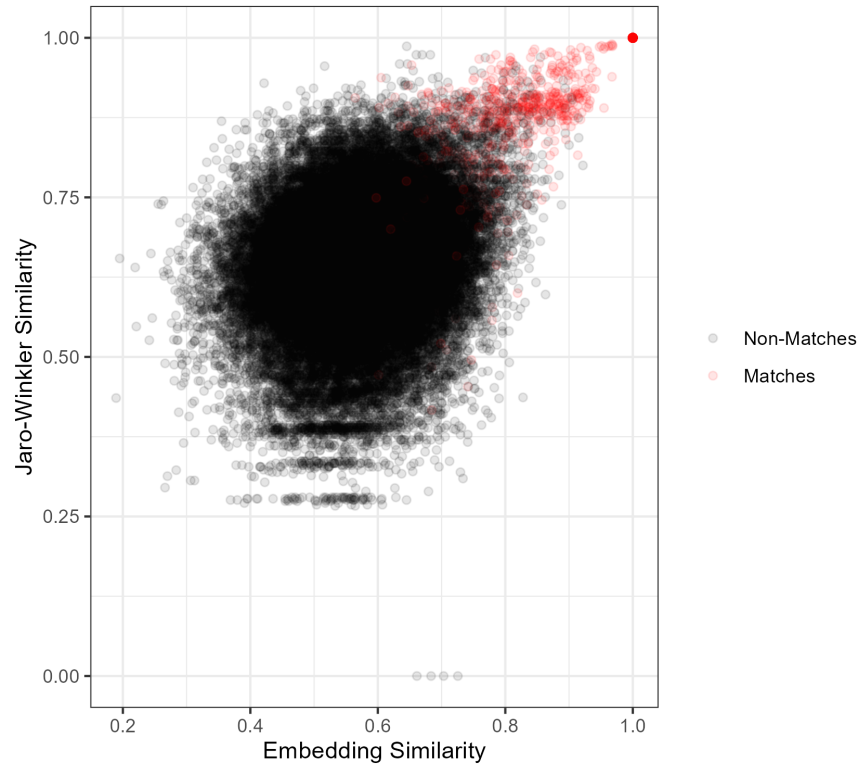


Figure A4: Similarity metrics for every within-block record pair in Application 1 (matching candidate names to the voter file). Red dots are confirmed matches and gray dots are non-matches.

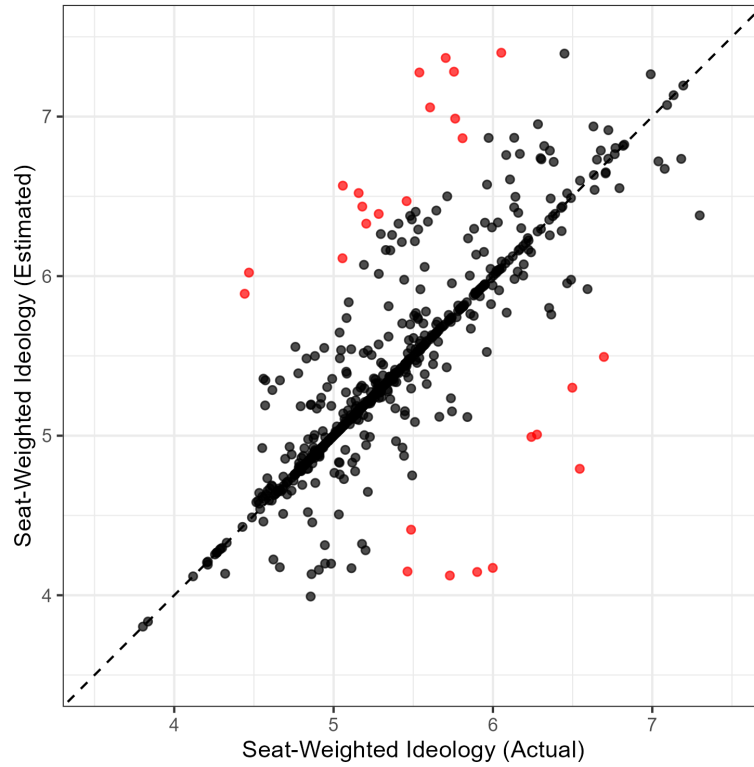


Figure A5: The actual seat-weighted ideology of each parliament in the ParlGov dataset (x-axis) plotted against estimated seat-weighted ideology estimated using labels and embeddings from open-source Mixtral 8x22B model (y-axis). Red points are those with absolute error greater than 1-point.



Figure A6: Estimated seat-weighted parliamentary ideology following merge (points) plotted over true values (lines), using labels and embeddings from open-source Mixtral 8x22B model.

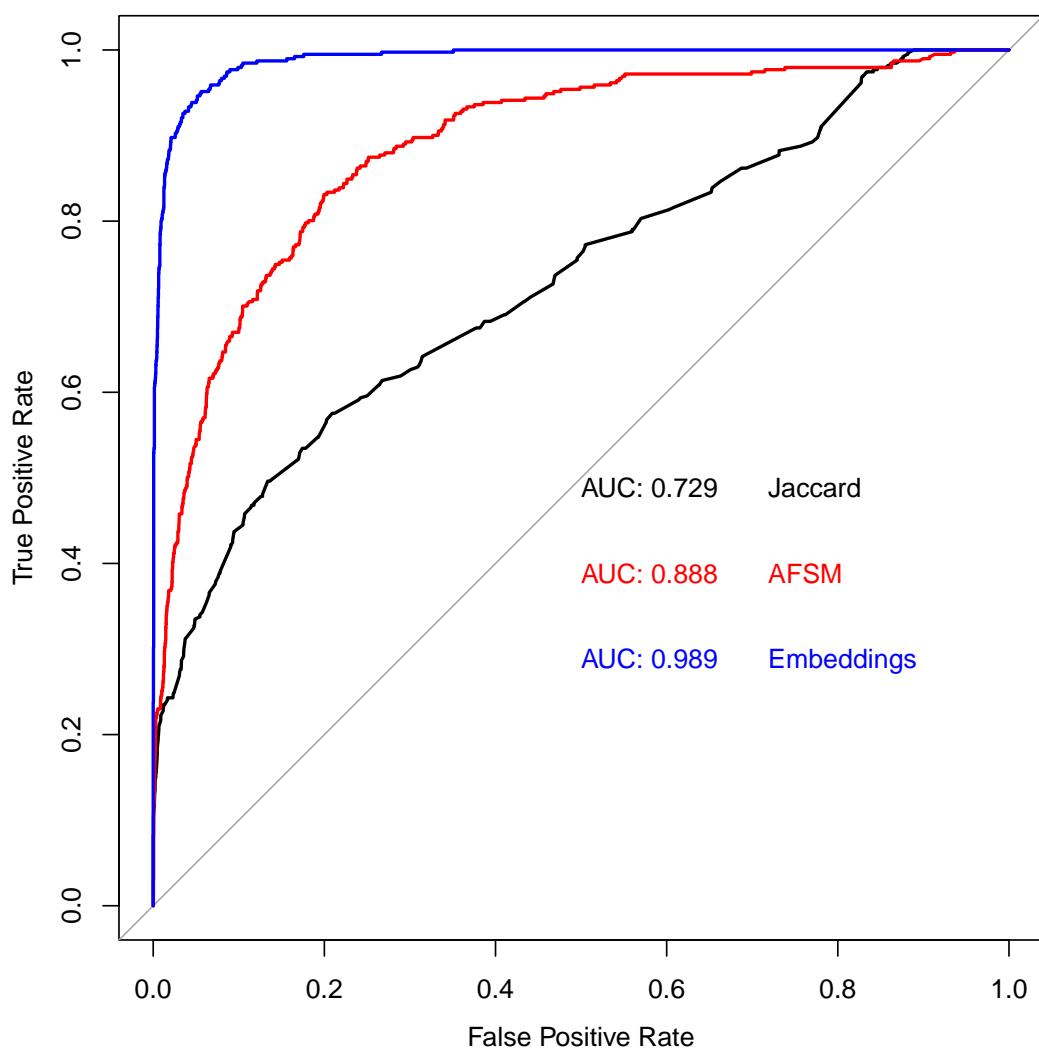


Figure A7: Receiver Operator Curves (ROC) for three fuzzy string similarity metrics on hand-labeled organization name pairs from Kaufman and Klevs (2022).

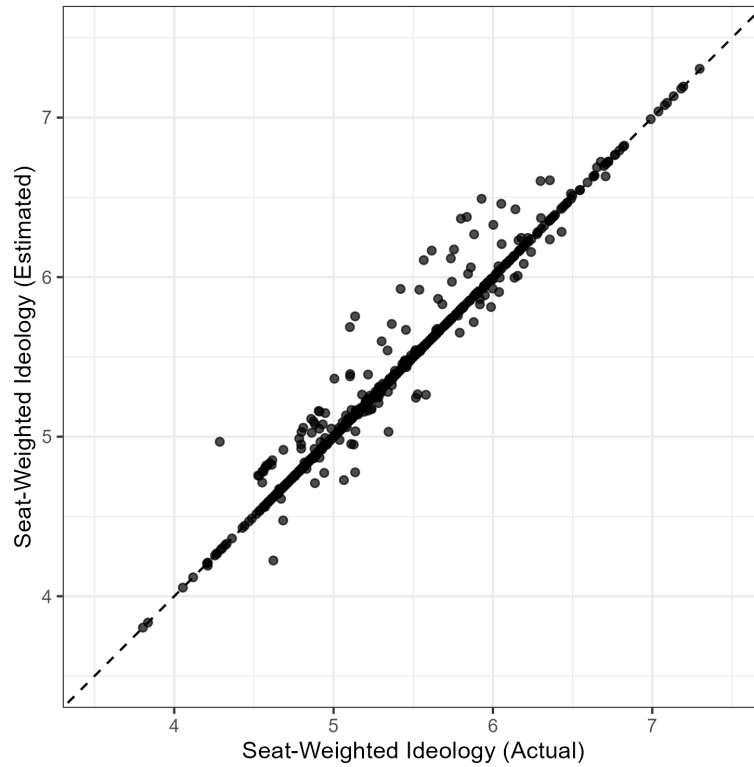


Figure A8: The actual seat-weighted ideology of each parliament in the ParlGov dataset (x-axis) plotted against estimated seat-weighted ideology following the probabilistic record linkage. Red points are those with absolute error greater than 1-point.

	Country	Precision	Recall
1	Austria	99.15	99.15
2	Belgium	93.89	100.00
3	Bulgaria	100.00	94.94
4	Croatia	100.00	98.81
5	Cyprus	94.83	100.00
6	Czech Republic	100.00	100.00
7	Denmark	89.98	100.00
8	Estonia	96.55	100.00
9	Finland	100.00	92.94
10	France	92.47	97.79
11	Germany	92.55	100.00
12	Greece	99.08	98.18
13	Hungary	100.00	100.00
14	Iceland	95.76	100.00
15	Israel	68.95	87.04
16	Italy	98.34	99.58
17	Japan	79.09	94.57
18	Latvia	100.00	98.53
19	Lithuania	97.75	100.00
20	Luxembourg	78.16	97.14
21	Malta	97.96	100.00
22	Netherlands	92.72	97.99
23	Norway	89.21	100.00
24	Poland	96.43	100.00
25	Portugal	100.00	100.00
26	Romania	92.68	97.44
27	Slovakia	100.00	100.00
28	Slovenia	98.94	98.94
29	Spain	92.35	94.94
30	Sweden	95.35	100.00
31	Switzerland	87.63	99.39
32	Turkey	100.00	79.25

Table A4: Precision and Recall for Multilingual Record Linkage Application By Country