DOI: 10.1111/tgis.12522

RESEARCH ARTICLE

WILEY Transactions in GIS

Machine learning innovations in address matching: A practical comparison of word2vec and CRFs

Sam Comber | **Daniel Arribas-Bel**

Department of Geography and Planning, University of Liverpool School of Environmental Sciences, Liverpool, UK

Correspondence
Sam Comber, Department of Geography and Planning, University of Liverpool School of Environmental Sciences, Roxby Building, 74 Bedford St, Liverpool L69 7ZT, UK.
Email: s.comber@liverpool.ac.uk

Funding information
Economic and Social Research Council

Abstract

Record linkage is a frequent obstacle to unlocking the benefits of integrated (spatial) data sources. In the absence of unique identifiers to directly join records, practitioners often rely on text-based approaches for resolving candidate pairs of records to a match. In geographic information science, spatial record linkage is a form of geocoding that pertains to the resolution of text-based linkage between pairs of addresses into matches and non-matches. These approaches link text-based address sequences, integrating sources of data that would otherwise remain in isolation. While recent innovations in machine learning have been introduced in the wider record linkage literature, there is significant potential to apply machine learning to the address matching sub-field of geographic information science. As a response, this paper introduces two recent developments in text-based machine learning—conditional random fields and word2vec—that have not been applied to address matching, evaluating their comparative strengths and drawbacks.

1 | INTRODUCTION

Address matching, the process of identifying pairs of records with a spatial footprint, is increasingly required for enriching data quality in wide-ranging, real-world applications. With government bodies, businesses and healthcare agencies drowning in an ever-increasing deluge of data, a competitive advantage exists in the analysis of integrated data sources as opposed to analyzing databases in isolation (Christen, 2012). Yet, in reality, most real-world

databases are noisy, inconsistent and replete with missing values. These issues complicate the integration of data. In fact, the acquisition of matched addresses is often key to spatially enabling data used for visualization or spatial data mining projects (Boulos, 2004). In the address matching context, while geospatial matching is directed by linking the geometric representations of spatial objects (Du, Alechina, Jackson, & Hart, 2017), spatial record linkage focuses on resolving text-based linkages between addresses.[1]

In the absence of unique identifiers that enable direct linking of data in relational database management environments, practitioners have traditionally relied on mathematical linkage techniques broadly divided by deterministic or probabilistic principles (Churches, Christen, Lim, & Zhu, 2002). While deterministic matching consists of generating hand-crafted rule bases for classification developed from specialist domain knowledge (Oliveira, Bierrenbach, Camargo, Coeli, & Pinhero, 2016), probabilistic linkage incorporates the varying distributions of a record's attribute values into the assignment of different weights for each field comparison. Weight assignment is related to the frequencies of value occurrences, with stronger weights given to matches for attributes upon which matching is less likely (Blanchette, DeKoven, De, & Roberts, 2013). For address matching, field comparisons might include comparing the street names of an address pair, with more common street names penalized by a lower weighting factor. In this way, resolving text-based postal addresses to the same address is a form of *geocoding*, where the quality of the match rate is intrinsically tied to the quality of the underlying reference data layer (Goldberg, 2011). Traditionally, address matching has focused on the probabilistic linkage approaches developed by the US Census Bureau in the 1970s (Jaro, 1984).

More recently, record linkage has been permeated by advances in machine learning. In this article, we focus on introducing two particular innovations into the address matching workflow: conditional random fields (CRFs) and word (address) embeddings. Before classification into address matches, input data requires segmentation into feature columns (Churches et al., 2002). The segmentation of postal addresses into attribute columns representing street numbers, street names or zip codes, for example, has been traditionally undertaken using hidden Markov models (HMMs). HMMs use statistical induction to predict, from possible arrangements of hypothetical states, the most likely arrangement to have produced the address sequence, and to then label each state by an attribute field (Christen, 2012). For addresses, labels might identify whether the present state represents a street number or street name. A recent innovation for text segmentation tasks has been the use of trained CRFs (Lafferty, McCallum, & Pereira, 2001). While HMMs assume the labeling of text sequences is statistically independent of previous outputs, CRFs are conditional by nature, meaning they assume no independence between output labels. Given that real-world text sequences such as addresses are represented by interaction and dependencies between words (e.g., zip codes are related to city names), it is reasonable to assume CRFs will perform well on a number of real-world text segmentation tasks.

A second innovation relates to the construction of so-called "comparison vectors" that are used for classifying records into matches and non-matches. Comparison vectors are created for each candidate record pair and contain several attributes that describe the text similarity of each pair (Christen, 2012). Traditionally, comparison vectors have been generated using string similarity metrics that measure the text distance between two address fields (e.g., the string similarity between two street names such as "Baker Street" and "Bakery Road"). Recently, however, advances from the natural language processing community demonstrate methods that map whole words and sentences to vectors in a continuous vector space. *Word2vec* (Mikolov, Chen, Corrado, & Dean, 2013) is one such method that maps semantically and syntactically similar words to nearby points in a vector space, encoding many linguistic patterns and regularities contained within the text. Such methods rely on a theory of language called the distributional hypothesis which states that words appearing in the same context purport similar meaning (Zellig, 1954). In the address matching context, one might hypothesize the word vectors generated for two semantically and syntactically similar postal addresses may be correctly resolved to a match.

While recent advances in machine learning have become adopted in the wider record linkage literature (Ektefa, Sidi, Ibrahim, Jabar, & Memar, 2011; Kopcke & Rahm, 2010; Nasseh & Stausberg, 2016), the address matching sub-field of geographic information science holds significant potential for the application of machine learning. In this article we explore how these advances can be integrated into the address matching workflow. In particular, we empirically evaluate

the performance of CRFs and word2vec in computing high-quality match rates between pairs of postal addresses. The remainder of this article is organized as follows. Section 2 introduces the data challenge. Section 3 motivates the methodology of the workflow. Section 4 presents the findings of the address matching methods applied. Section 5 concludes.

2 | DATA

Our comparison relies on a set of addresses previously matched by the Local Data Company (LDC) (Singleton, 2015). This provides a ground truth of address pairs with a known match status, which allows us to evaluate the performance of the linkage methods. These address pairs were obtained from a previous round of matching between non-domestic addresses of the LDC and Valuation Office Agency (VOA) databases. In particular, these address pairs reflect matches between LDC records of high-street shops and commercial addresses contained in the VOA 2010 rating list (VOA, 2017). A description of the address fields for the LDC and VOA addresses that were segmented by a method we introduce later, the conditional random fields, is introduced in Table 1, to familiarize the reader with components of a structured address string.

Crucially, the matched set of LDC to VOA addresses contains 110,742 pairs that resolve to the same address. This matched set is augmented with 934,150 synthetic non-matched pairs that are generated with the Freely Extensible Biomedical Record Linkage (FEBRL) (Christen & Churches, 2005) data set generator. This works by creating variants of the matched addresses with different error characteristics introduced to the data, meaning the models learn the representations of non-matched addresses (Christen, 2012). Thus, for each address field in Table 1, with the exception of zip code, we introduce error characteristics to the data. A demonstration of how the synthetic non-matches are generated is given in Table 2, where three examples of records from the LDC and VOA database are mutated with different error characteristics. In particular, we set the probability of a missing field as proportional to the number of missing fields in the matched addresses. Moreover, we set the maximum number of modifications per address field and per address string to one, also testing a scenario where we increase the number of modifications to nine later on. These modifications introduce a probability for a character in the address field to be randomly inserted, deleted, substituted or transposed. Importantly, the match status of these synthetic non-matches is always set to false, meaning our machine learning techniques learn the representations of non-matched addresses for highly nuanced cases. To prepare the data for segmentation, we append all address fields from the LDC and VOA data sets into a comma-separated address string (e.g., "Home Bargains, 28, Church Way, Bradford") while keeping the zip code separate for reasons we explain immediately below.

TABLE 1 CRF parser label tags and descriptions identified for the LDC and VOA addresses

Tag	Description	Example
House	Venue or business name ascribed to the address	Automotive Solutions
Number	Street-facing building number or apartment number	43
Unit	A secondary unit designator that identifies an office, unit or apartment	4a
Level	Expression signifying a floor number	Ground Floor
Street	Identifying name given to a street	Paradise Street
Suburb	Unofficial neighborhood name	Ropewalks
City	Any human settlement such as a metropolis, city, town or village	Liverpool
District	Second-level administrative division	North West
State	First-level administrative division	England
Zip code	Postal code used for mail sorting	L3 5TB

Note. Tags are aligned to address fields of the OpenCage (2018) address formatting library.

TABLE 2 FEBRL data set generator (Christen & Churches, 2005) demonstration for example records from the LDC and VOA database

ID	House	Number	Unit	Level	Street	Suburb	City	District	State
1-original-LDC	Kwik Fit	67–69	–	–	Whiteladies Road	–	Bristol	South West	
1-duplicate-1	–	41	Unit A	–	White Avenue	–	Bristol	South West	England
1-duplicate-2	Fitness Quick	64–66	–	–	Whiteladies Road	–	Bristol	–	
2-original-VOA	Thomson	–	Unit 19c	–	Teeside Retail Park	Goodwood Square	Stockton-on-Tees	North East	–
2-duplicate-1	–	5	Unit 1a	–	Teeside Retail Park	Goodewood Square	Stockton-on-Tees	North East	
2-duplicate-2	–	–	Unit 5b	–	Teaside Retail Park	Goodwood Square	Stockton-on-Tees	North East	
3-original-VOA	Body Shapers	–	72c	First	Ridgeway Street	–	Plymouth	South West	–
3-duplicate-1	–	–	8a	Second	Ridgeway Street	–	Plymouth	South West	–
3-duplicate-2	Bdoy Shapres	–	72e	–	Longmore Road	–	Plymouth	South West	–

Our next step introduces *blocking* for each method to increase the computational tractability of the linkage task. On a Dell Precision Tower 7000 series with 60 GB RAM and multi-core processor, the computational cost for the comparison of each address in the LDC and VOA data sets is substantial. This is because the number of address comparisons without blocking is a function of the Cartesian product of both data sets, which has quadratic complexity $O(n^2)$. So, for example, if the LDC and VOA data sets both contained just 10^4 records, the linkage task requires 10^8 comparisons, which becomes computationally non-trivial. To remedy this, we introduce blocking to partition the set of all possible address comparisons between the LDC and VOA databases to within mutually exclusive blocks (Newcombe & Kennedy, 1962). Now, if we let b equal the number of blocks, we are left with n/b addresses per partition on average, which reduces the complexity to $O(n^2/b)$ (Christen, 2012). This means the linkage task becomes tractable even on low-performance machines, as the linkage within each partition can be processed sequentially or, alternatively, in parallel if the user has access to a multi-core machine. Therefore, in each of the following methods, we use the zip codes of postal addresses as a blocking key. This reduces the number of address pair comparisons to within 39,855 zip code "blocks," with the distributional characteristics of these partitions displayed in Figure 1. In our case, the near uniform frequency distribution of the zip code blocks and completeness of the zip code attribute means it is a sensible choice as a blocking key. Yet, for different address databases where the zip code column is replete with missing values, an alternative attribute should be considered as a blocking key, which is an empirical decision to be motivated by the characteristics of the databases' attribute columns.

One potential issue with using zip codes as a blocking key is the existence of typographic errors in the spelling. In our case, this is pertinent because while validation checks are employed by the LDC, the recording of commercial addresses is undertaken by teams of surveyors, and is therefore susceptible to human error. To account for this, we explore *sorted neighborhood blocking*. We sort together the LDC and VOA data sets using the zip code as a sorting key value while restricting address comparisons to records within a window of fixed dimension, $w = 5$. As the window slides over the sorted zip codes, the identification of matches and non-matches is restricted to candidate address pairs within this window of fixed size (Cibella & Tuoto, 2012). This means that the technique is highly sensitive to lexicographic order which, in our case, is advantageous because we create candidate address

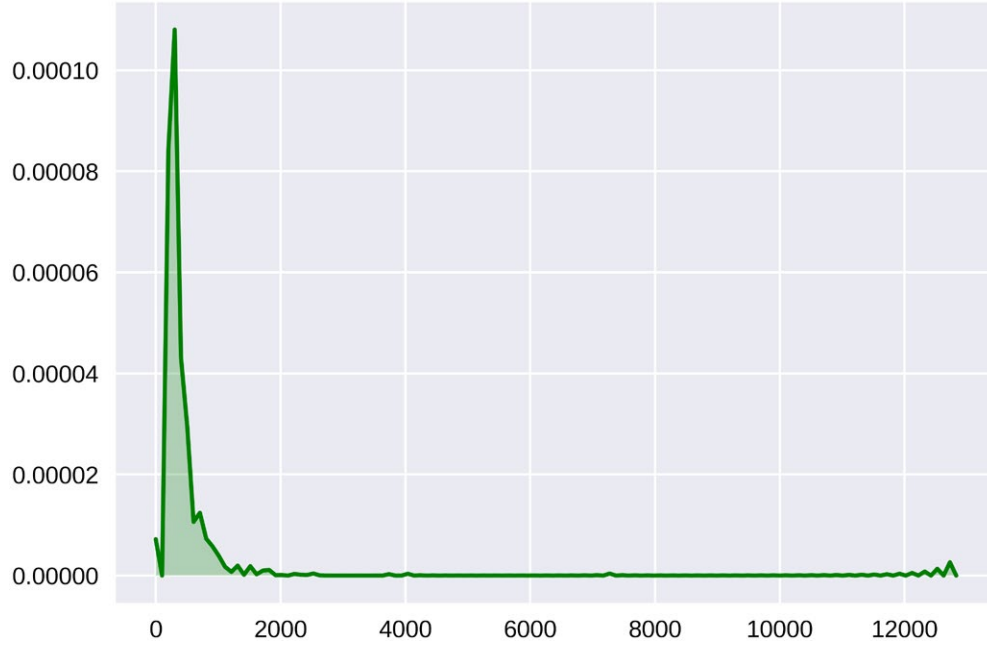

FIGURE 1 Distributional characteristics of the address block partitions (n = 39,855)

pairs from misspelt zip codes (e.g., the comparison between addresses with sorted zip codes "L3 5TA" and "L3 5TB" that pertain to the same address). However, a limitation of sorted neighborhood blocking is that misspellings of the first character in the blocking key can erroneously direct addresses to incorrect zip code blocks. Q-gram blocking is an alternative approach to generating partitions of zip code blocks, and converts blocking keys to a list of bigrams (Q = 2). The zip code "SW11 9LU", for example, generates a bigram list ["SW","W1","11","19","9L","LU"], from which *k* sub-lists of length *k*−1 are generated recursively to create variations of the zip code (Christen, 2012). This continues up to a minimum threshold for the number of bigrams in the sub-list. Following the example, this might create blocking key values such as "W1 9LU", "SW1 9LU" and "SW19 LU", meaning that the same address is likely to be inserted into several different zip code blocks. While the recursive generation of sub-lists is computationally expensive, the advantage of Q-gram blocking is that it overcomes typographic error in the characters of the blocking key. In our case, as the results presented below show, given the zip code attribute was profiled as generally being of high quality, the choice of blocking mechanism was less fundamental. Nevertheless, we introduce additional blocking techniques to provide instructional guidance to researchers interested in replicating our methodology on their own data sets, where attributes of the blocking key values might be less clean.

3 | METHODS

3.1 | Conditional random fields

One of the principal challenges in obtaining high-quality match rates is the conversion of raw data into a structured, usable format for comparison. For postal addresses, this involves parsing address sequences into feature columns. Take, for example, a canonical address of the form "3B Records, 5 Slater Street, Liverpool L1 4BW". Our objective is to segment (or label) this address into appropriate columns for business name, property number, street name, city and zip code. To use a hidden Markov model (Baum & Petrie, 1966) for segmentation would be to assign a joint probability to the observation sequence where the labeling of address elements is independent of previous labels (Churches et al., 2002). This means, following the example above, that "3B" could be incorrectly classified as a property number, whereas it actually completes the business name "3B Records". Importantly, "3B" will now be considered a property number, and that (alternative) fact will be used to classify the next token, "Records". This leads to an erroneous sequence of label predictions.

In real-world text sequences such as addresses, the probability of a transition between labels might depend not just on the current address element, but also on past and future elements. For this reason, conditional random fields (Lafferty et al., 2001) are more suited to addressing segmentation tasks. Principally, this is because CRFs negate what is known as the *label bias problem*: "transitions leaving a given state to compete only against each other, rather than against all transitions in the model" (Lafferty et al., 2001). When the CRFs has parsed "3B" and reaches the second token, "Records", the model scores an *l*×*l* matrix where *l* is the maximum number of labels that can be assigned by the model. In *L*, element l_{ij} reflects the score for the probability of the current word being labeled *i*, and the previous word labeled *j* (Diesner & Carley, 2008). Returning to the example, when the parser gets to the *actual* property number, "5", the highest score in the matrix indicates the current label should be revised to a property number, and the previous label to a business name. Below, we provide an illustrative example of an erroneous sequence of labels predictions that hypothetically may have been segmented with an HMM.

3B	*Records*	*Slater Street*	*Liverpool*	L1 4BW
NUMBER	STREET	SUBURB	CITY	ZIP CODE

In the CRFs, prediction of the most likely sequence of labels uses a reversible highest *scoring* path. This is known as Viterbi inference (Viterbi, 1967), and leads to a sequence of labels with the highest likelihood.

3B Records	*5*	*Slater Street*	*Liverpool*	L1 4BW
HOUSE	NUMBER	STREET	CITY	ZIP CODE

With the CRFs model, the previously raw and unstructured address will now be correctly segmented into the following feature columns that can be used as a basis for classifying records into matches and non-matches:

House:	3B Records
Number:	5
Street:	Slater Street
City:	Liverpool
Zip code:	L1 4BW

In our case, address segmentation is undertaken using the *Libpostal* C library (Barrentine, 2018) that trains a CRFs model on addresses sourced from OpenStreetMap (OSM) data. This means we apply a pre-trained address segmentation model to each raw address string, setting the country code of the parser to "GB" (for Great Britain) so the software recognizes it is segmenting UK addresses. Libpostal's *parse_address* command will then label the address sequence into features columns, if they exist, for the fields in Table 1. To empirically evaluate the performance gain, we also introduce an HMM^2 alongside the CRFs parser. Once feature columns consisting of address elements have been obtained for every LDC and VOA address, a *comparison vector* is constructed for each candidate address pair. Comparison vectors contain several attributes that describe the text similarity between each feature column (Christen, 2012). In our case, each element of this comparison vector contains the Jaro–Winkler string similarity between each address field, with exception of zip code, in Table 1. Briefly, the Jaro–Winkler distance calculates the minimum number of single character transpositions required to convert one string into another, also increasing the similarity when the first few characters are the same (Herzog, Scheuren, & Winkler, 2007). We motivate the decision to use Jaro–Winkler as our string comparison function because previous findings show it performs best on attributes containing named values (e.g., property names, street names, or city names) (Christen, 2012; Yancey, 2005). If a given address field is missing, the Jaro–Winkler similarity between the pair of address fields is set to zero. These comparison vectors for each address pair are the basis of a binary classification for classifying whether address pairs resolve to matches or non-matches. The general idea is that the more similar two addresses are, as described by Jaro–Winkler similarity, the higher the likelihood that they resolve to the same address.

Our classification approach is supervised, meaning we use our training data of known true match and true non-match status generated in Section 2 to evaluate the outcome of our address matching exercise. By training a classifier, we allow the model to learn the nuances of matched and non-matched addresses. This means that, after training, we can test whether unseen address pairs for which the match status is known correctly resolve to matches and non-matches, allowing us to evaluate the performance of our linkage techniques. Thus, once comparison vectors have been generated for each training record, we introduce several classifiers to facilitate the linkage into matches and non-matches. In particular, two ensemble methods for classification, a random forest (Breiman, 2001) and gradient boosted classifier known as XGBoost (Chen & Guestrin, 2016), are trained alongside a logistic regression model. We motivate our use of ensemble classifiers for one key reason. Our comparison vectors are embedded into a nine-dimensional vector space. This means each dimension reflects one of the nine Jaro–Winkler similarities between each address field in Table 1, with the exception of zip codes which are used for blocking. When partitioning matches from non-matches in this 9-dimensional vector space, while the logistic model searches for a linear decision boundary, the multiple decision trees of the ensemble methods partition the vector space into half spaces by using axis-aligned linear decision boundaries (Efron & Hastie, 2016). This has the net effect of a nonlinear decision boundary, which is desirable if the comparison vectors cannot be accurately separated into matches and non-matches by a single hyperplane. From here, the classifiers are trained using *k*-fold

cross-validation where $k = 10$, which we explain in Section 4, and are evaluated against metrics commonplace in machine learning such as precision and recall (Christen, 2012).

3.2 | Word embeddings

In our second approach, we augment the use of CRFs with so-called "word embeddings," which is the name given to the vector representations of words. Vector space models *embed* words in a continuous vector space, where words with similar syntactic and semantic meaning are mapped, or embedded, to nearby points (Mikolov et al., 2013). Such methods leverage the distributional hypothesis of language which states that "words which are similar in meaning occur in similar contexts" (Rubenstein & Goodenough, 1965). One such method is word2vec (Mikolov et al., 2013), a neural probabilistic language model whose training objective is to find word vector representations that are good at predicting the surrounding words in a text-based sentence or document.[3] In our case, we hypothesize that learning high-dimensional vectors from postal addresses may be used to match addresses that resolve to the same geographic location despite irregularities in the text. In practice, we train *gensim*'s (Řehuřek & Sojka, 2010) implementation of word2vec on 29.6 million parsed postal addresses from the UK Postcode Address File (PAF) database (PAF, 2018). Learning word embeddings using word2vec requires setting the dimensionality of the vectors, so the training phase begins by randomly initializing each address field component with 100 real numbers. This means each parsed address field is represented by an array of numbers of length 100 which, as an example, can be represented as: $[0.32 \quad 0.28 \quad \ldots \quad 0.01 \quad 0.58] \in \mathcal{R}^{100}$. By feeding successive address fields into the model, the real numbers of each word vector are updated so that words sharing the same context are mapped closer together in the vector space. To build intuition for this idea, we employ Figure 2, where the *t*-distributed stochastic neighbor embedding (t-SNE) (van der Maaten & Hinton, 2008) dimensionality reduction technique is applied to the top 10 closest vectors to an address

FIGURE 2 t-SNE visualization demonstrating the top 10 closest vectors to "Halifax PLC" in a two-dimensional vector space

field for property name, "halifax plc". In this two-dimensional vector space, the closeness of word vectors, measured by cosine similarity, represent words that share closer semantic and syntactic meaning. "Halifax PLC", for example, is a bank, and interestingly the word vector generated for it is embedded nearby to businesses that have a financial remit, "Natwest", "TSB Bank" and "Barclays Bank PLC", for example.

Under this approach, instead of using the Jaro–Winkler similarity between the address fields, we augment the linkage task by comparing word vectors generated by word2vec for the address fields segmented by the CRFs model. Thus, after training a word2vec model on PAF addresses, for every LDC and VOA address we are able to obtain a 100-dimensional vector for each address field. The postal address, "5 Myrtle Street, Liverpool", for example, contains three address fields (a street number, a street name, and a city name) for which we obtain vectors. Similarly to our first approach, we construct a comparison vector, but this time each element is the *cosine similarity* between the word vectors constructed from the address fields parsed by the CRFs model. In cases where address fields are missing, we set the cosine similarity to zero, which implies orthogonality or linear independence between the vectors under comparison. The decision to choose cosine similarity as the criterion for measuring similarity between address fields is because it has favorable qualities in capturing the semantic closeness of word vectors (McInnes & Pedersen, 2013). As before, we train a random forest, XGBoost and logistic regression model on the comparison vectors and associated match status labels using *k*-fold cross-validation to evaluate the linkage performance.

4 | RESULTS

To evaluate the performance of the HMM and CRFs alongside the CRFs augmentation with word2vec, we use address pairs for which the match status is known (as discussed in Section 2). The results of these methods are highlighted in Table 3, and are benchmarked by evaluation metrics known as *recall* and *precision*. Recall measures the proportion of address pairs that should have been classified, or recalled, as matched (Christen, 2012). The precision (or, equivalently, the positive predictive value) calculates the proportion of the matched address pairs that are classified correctly as true matches (Christen, 2012). To minimize over-fitting our supervised models, we introduce *k*-fold cross-validation, where *k*=10, meaning the training data is split into ten disjoint groups. In each split, the classifier is trained and tested on these subsets of address pairs, with the resulting recall and precision averaged across the groups. This means that for each group we have a randomized training and testing set split by 75% and 25%, respectively.

TABLE 3 Recall and precision evaluation metrics for the HMM, CRFs, and CRFs augmented using word2vec

Method	Precision	Recall
HMM		
Logistic	0.738	0.459
Random forest	0.944	0.696
XGBoost	0.959	0.688
CRFs		
Logistic	0.933	0.820
Random forest	0.940	0.918
XGBoost	0.955	0.902
CRFs–word2vec		
Logistic	0.870	0.687
Random forest	0.933	0.874
XGBoost	0.950	0.870

Note. Results are 10-fold cross-validated using 25% of the data for testing within each fold.

To begin interpretation, we first turn our attention to the baseline HMM that we use as a point of comparison for the machine learning techniques we introduce. Consistent with our earlier motivations, when address fields are parsed with the CRFs model, they outperform the HMM. This is shown by the lower recall values retrieved by each of the classifiers using the HMM technique. Interestingly, the precision values of the HMM and CRFs techniques are broadly consistent. This implies that, of the total number of matches returned, both techniques perform well at partitioning true positives from false positives, but the CRFs classify a larger proportion of matches, as shown by the higher recall value. This finding suggests that, unlike the HMM, the reversible sequence of labeling introduced by the CRFs leads to higher-quality match rates. We now turn to the supervised classifiers that are trained on the comparison vectors built using the CRFs model. An interesting facet of tree-based models is that the feature importances can be recovered (Hastie, Tibshirani, & Friedman, 2009). The "importance" of different features, or, equivalently, address fields, to the match classification is visualized by the red bars in Figure 3, along with the inter-trees variability. Figure 3 is consistent with conventional wisdom, as it indicates that street name and house number are the most important features that are used when resolving candidate pairs of addresses to a match. To visualize the absolute numbers of matches, we provide a confusion matrix in Figure 4 for the random forest trained with address pairs parsed by the CRFs. In Figure 4, the top left quadrant shows true negatives, top right shows false positives, bottom left shows false negatives, and bottom right shows true positives. Briefly, true positives are address pairs labeled as matches that are true matches; false positives are address pairs mislabeled as matches; true negatives are records classified as non-matches which are true non-matches; and false negatives are addresses classified as non-matches but are actually true matches (Christen, 2012).

From Table 3 it is clear the ensemble learners offer slight improvement over the logistic model in returning a larger fraction of true positives among all returned "matches". This is shown by the marginally higher precision

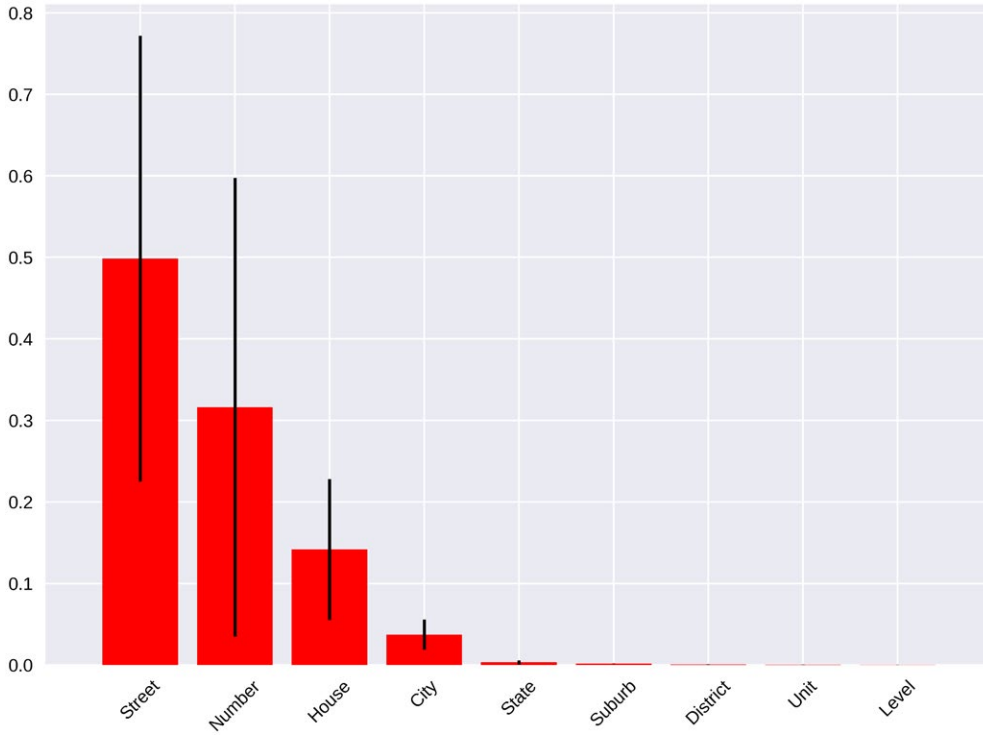

FIGURE 3 Feature importances of address fields from Table 1 to matching outcomes. Importances are given for the random forest model trained on address fields segmented by the CRFs

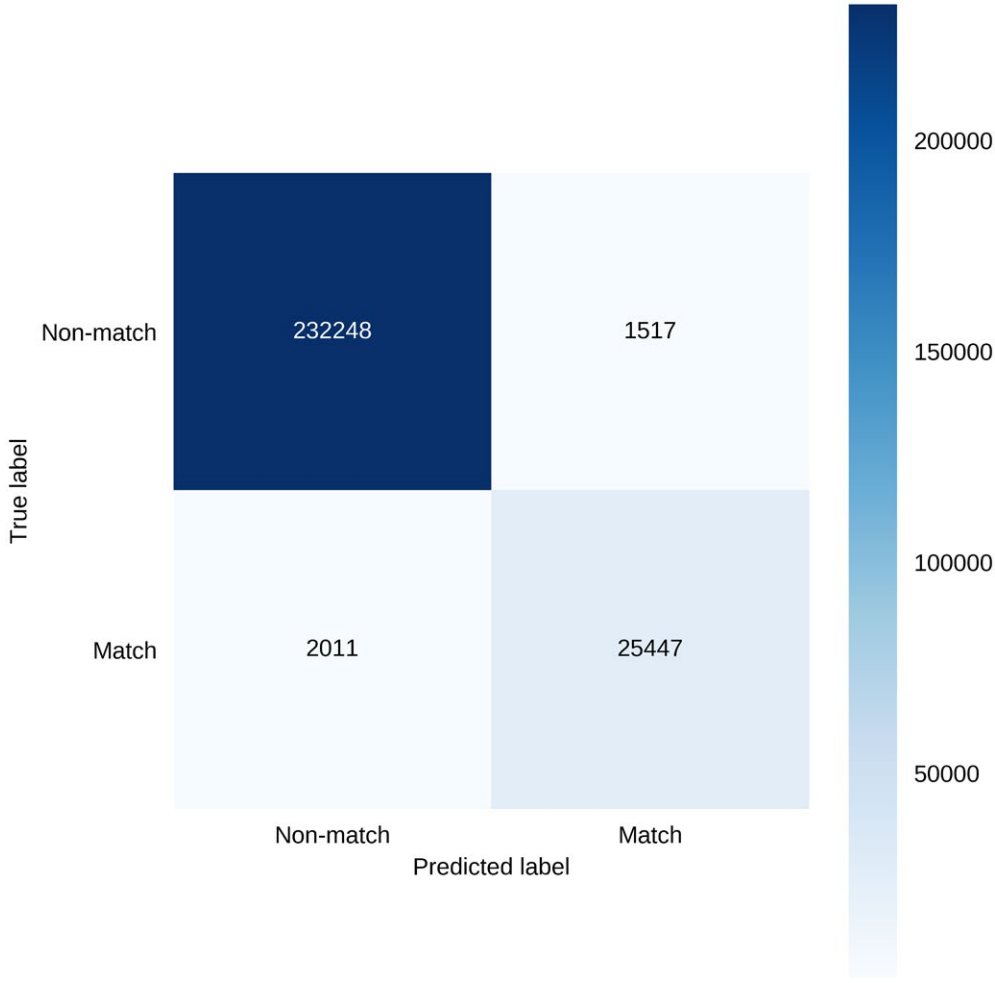

FIGURE 4 Confusion matrix for true positives, false positives, true negatives and false negatives retrieved by the random forest classifier for CRFs segmentation

value for the random forest, which we complement by displaying a precision–recall curve in Figure 5. The green line in the figure suggests that while the random forest performs well at classifying true matches from the matches it returns, it performs less well at retrieving all matched instances. Between the two ensemble approaches, XGBoost classifies the highest number of true matches correctly, with only marginal differences in recall, or retrieval of relevant address pairs, between the two. Presumably, the ensembles perform better because the vector space for classifying address pairs is not linearly separable, and requires a nonlinear decision boundary to partition matches from non-matches to a high degree of accuracy.

Next, we turn our attention to the CRFs method that is augmented with the use of word2vec for address field comparisons. While the first method uses Jaro–Winkler similarity to assess string distance between address fields parsed by the CRFs model, our second approach augments the first by replacing Jaro–Winkler similarity with cosine similarity between word vectors learnt from the parsed address fields. Overall, this augmented approach is highly competitive with the first approach. If, for example, we take the XGBoost findings from Table 3 as a point of comparison, the precision and recall values decrease by 0.005 and 0.032, respectively, in the augmented approach when compared to the first approach. This raises the question of what advantage CRFs augmentation with word-2vec yields. The principal advantage is that it does not commit the user to the biases of a particular string distance

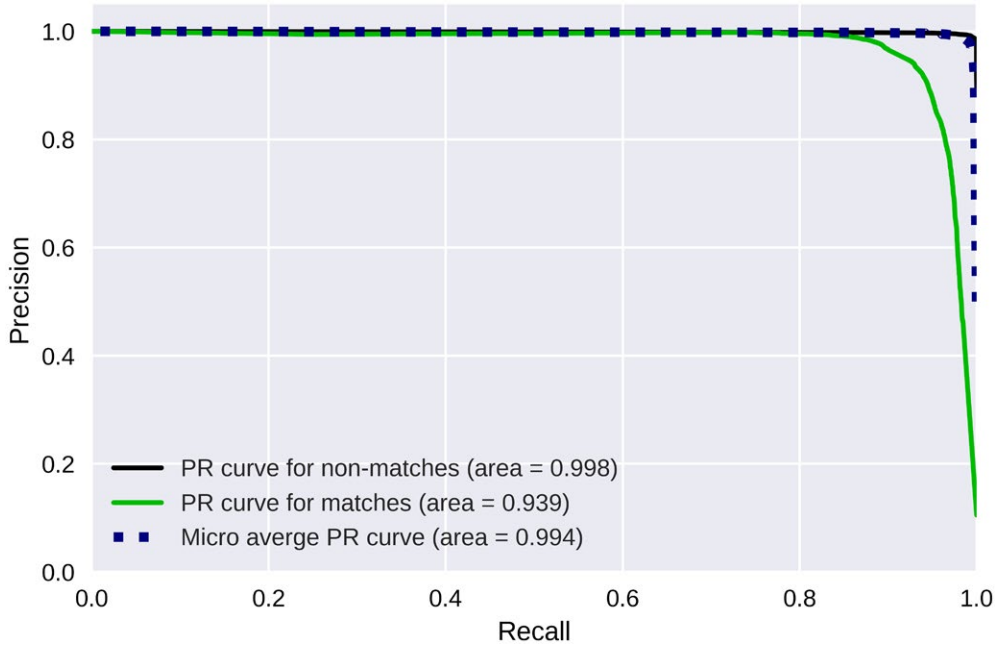

FIGURE 5 Precision–recall (PR) curve for the random forest classifier trained on addresses segmented by the CRFs model

function. Jaro–Winkler similarity, for example, is highly tuned to named attribute values, and so the use of word vectors provides a more generalizable approach for matching address fields. In all, the findings support our theoretical motivations from Section 3, and imply that the word vectors obtained for the address fields perform well in capturing the syntactic structures and regularities required to resolve address pairs to a match.

Finally, we tweak two components of our empirical design to evaluate the robustness of our main findings. In doing so, we evaluate any substantive change in the match performance for our preferred specification, the XGBoost classifier, trained on comparison vectors generated by the CRFs model. Firstly, we adjust the probabilities for introducing modifications to the synthetic non-matched addresses generated by FEBRL. Here, we apply two scenarios: in one, we set the probability of a missing field equal to the proportion of missing fields in the matched addresses, while setting the maximum modification per field and maximum modification per address to one; and in the other, we maintain the same probability for missing fields, but increase the maximum number of modifications per field and maximum modifications per address to nine in both cases. Modifications refer to insertions, deletions and transpositions of characters in the address field. These scenarios are introduced to evaluate the extent to which the classifier performance changes as we degrade the quality of our non-matches in the training data. As expected, the precision value increases marginally from 0.955 to 0.973 as we increase the number of error modifications to the non-matched addresses. This is intuitive as increasing error in the non-matches means the address pairs become more dissimilar. Therefore, it becomes easier for the classifier to disambiguate between matches and non-matches as the nuances between non-matched address pairs become less pronounced. Our second tweak tested the change of blocking mechanism from standard blocking to sorted neighborhood blocking and Q-gram blocking. This was applied to better handle cases of misspelt zip codes, which is problematic because misspellings could allocate addresses to incorrect zip code blocks. Despite the empirical motivations to alternate the blocking mechanism, we found the results of our main findings were invariant to which blocking technique was applied. Presumably this was because zip codes of the LDC and VOA addresses are of high quality, as the

VOA addresses originate from an official UK government source and the LDC have a business case in maintaining accurate, high-quality zip codes.

5 | CONCLUSIONS

Often the biggest problem when faced with spatial data is accessing it. Address matching resolves text-based address sequences to matches, integrating disparate sources of data that would otherwise remain in isolation. In this article we evaluated the performance of two recent machine learning techniques for linking address pairs where the match status was already known. Our first technique, the CRFs approach, focused on segmenting whole postal addresses into address fields, which became the basis for constructing comparison vectors for every candidate address pair in the data set. Once obtained, supervised classifiers were applied to partition the comparison vectors of address pairs into matches and non-matches. In all, the classifiers trained using addresses segmented by the CRFs achieved a precision of up to 0.955, with the ensemble learners outperforming the logistic model. This was likely due to the improved fit of a nonlinear decision boundary to the underlying vector space.

Our second approach augmented the first by replacing the string similarity metric we used to compare parsed address fields from the CRFs model with a comparison between word vectors. These were generated using a technique called word2vec, which sought to embed semantically and syntactically similar address fields to nearby locations in the vector space, with the expectation that vectors embedded *nearer* together could be used to match address fields. As before, we used supervised classifiers to facilitate the linkage, which resulted in a precision of up to 0.950. This value implied the vectors obtained for the address fields performed successfully at encoding word relationships, patterns and regularities that are required to facilitate accurate linkage between address pairs. In synthesis, the main implications of this article point to the utility of CRFs, and their augmentation with word2vec, for the accurate segmentation of addresses. These steps are preconditions for constructing high-quality comparison vectors that can be used to accurately classify address pairs into matches and non-matches.

NOTES

[1]Environmental health studies, for example, rely on spatial record linkage to determine whether individuals in residential locations live within defined zones of exposure to hazardous environments (Baldovin et al., 2015; Cayo & Talbot, 2003; Reynolds et al., 2003).

[2]Our HMM is trained on the same OSM addresses for the UK as Libpostal. They are obtained by filtering the "great-britain-latest.osm.pbf" file available from Geofabrik (2018). Filtering is performed using the *Osmosis* command line application (OpenStreetMap, 2018) that allows us to distill addresses from the entirety of OSM data in the file. Therefore, we use Osmosis to filter by the following tags: "addr:housename", "addr:housenumber", "addr:street" and "addr:postcode". Implementation for the HMM model is provided by a script available from FEBRL (Christen & Churches, 2005) which tags free-text address sequences from lookup tables before rearranging the tags to the most likely sequence of address fields.

[3]For details of technical implementation, the reader is referred to Mikolov et al. (2013).

REFERENCES

Baldovin, T., Zangrando, D., Casale, P., Ferrarese, F., Bertoncello, C., Buja, A., … Baldo, V. (2015). Geocoding health data with geographic information systems: A pilot study in northeast Italy for developing a standardized data-acquiring format. *Journal of Preventive Medicine & Hygiene*, *56*, 88–94.

Barrentine, A. (2018). *Libpostal*. Retrieved from https://github.com/openvenues/libpostal

Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, *37*, 1554–1563.

Blanchette, C., DeKoven, M., De, A., & Roberts, M. (2013). Probabilistic data linkage: A case study of comparative effectiveness in COPD. *Drugs in Context*, *2013*, 212258.

Boulos, M. N. K. (2004). Towards evidence-based, GIS-driven national spatial health information infrastructure and surveillance services in the United Kingdom. *International Journal of Health Geographics*, *3*, 1–50.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Cayo, M. R., & Talbot, T. O. (2003). Positional error in automated geocoding of residential addresses. *International Journal of Health Geographics*, *2*, 1–10.

Chen, T., & Guestrin, C. (2016). *XGBoost: A scalable tree boosting system*. arXiv:1603.02754.

Christen, P. (2012). *Data matching: Concepts and techniques for record linkage, entity resolution, and duplicate detection*. New York, NY: Springer.

Christen, P., & Churches, T. (2005). *FEBRL, version 0.3.1*. Retrieved from http://users.cecs.anu.edu.au/ Peter.Christen/Febrl/febrl-0.3/febrldoc-0.3/manual.html

Churches, T., Christen, P., Lim, K., & Zhu, J. X. (2002). Preparation of name and address data for record linkage using hidden Markov models. *BMC Medical Informatics and Decision Making*, *2*, 1–9.

Cibella, N., & Tuoto, T. (2012). Statistical perspective on blocking methods when linking large data-sets. In A. Di Ciaccio, M. Coli, & J. M. Angulo Ibanez (Eds.), *Advanced statistical methods for the analysis of large data-sets* (pp. 81–89). Berlin, Germany: Springer.

Diesner, J., & Carley, K. M. (2008). Conditional random fields for entity extraction and ontological text coding. *Computational and Mathematical Organization Theory*, *14*, 248–262.

Du, H., Alechina, N., Jackson, M., & Hart, G. (2017). A method for matching crowd-sourced and authoritative geospatial data. *Transactions in GIS*, *21*, 406–427.

Efron, B., & Hastie, T. (2016). *Computer age statistical inference: Algorithms, evidence, and data science*. Cambridge, UK: Cambridge University Press.

Ektefa, M., Sidi, F., Ibrahim, H., Jabar, M., & Memar, S. (2011). A comparative study in classification techniques for unsupervised record linkage model. *Journal of Computer Science*, *7*, 341–347.

Geofabrik. (2018). *OpenStreetMap data for Great Britain*. Retrieved from https://download.geofabrik.de/europe/great-britain/

Goldberg, D. W. (2011). Improving geocoding match rates with spatially-varying block metrics. *Transactions in GIS*, *15*, 829–850.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). Random forests. In T. Hastie, R. Tibshirani, & J. Friedman (Eds.), *The elements of statistical learning: Data mining, inference and prediction* (2nd ed., pp. 587–604). Berlin, Germany: Springer.

Herzog, T. N., Scheuren, F. J., & Winkler, W. E. (2007). *Data quality and record linkage techniques*. New York, NY: Springer.

Jaro, M. (1984). *Record linkage research and the calibration of record linkage algorithms (Statistical Research Division Report Series No. Census/SRD/RR-84/27)*. Washington, DC: U.S. Bureau of the Census.

Kopcke, H., & Rahm, E. (2010). Frameworks for entity matching: A comparison. *Data & Knowledge Engineering*, *69*, 197–210.

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In C. E. Brodley & A. P. Danyluk (Eds.), *Proceedings of the 18th International Conference on Machine Learning* (pp. 282–289). San Francisco, CA: Morgan Kaufmann.

McInnes, B. T., & Pedersen, T. (2013). Evaluating measures of semantic similarity and relatedness to disambiguate terms in biomedical text. *Journal of Biomedical Informatics*, *46*, 1116–1124.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient estimation of word representations in vector space*. arXiv:1301.3781.

Nasseh, D., & Stausberg, J. (2016). Evaluation of a binary semi-supervised classification technique for probabilistic record linkage. *Methods of Information in Medicine*, *2*, 136–143.

Newcombe, H., & Kennedy, J. (1962). Record linkage: Making maximum use of the discriminating power of identifying information. *Communications of the ACM*, *5*, 563–566.

Oliveira, G., Bierrenbach, A., Camargo, K., Coeli, C., & Pinhero, R. (2016). Accuracy of probabilistic and deterministic record linkage: The case of tuberculosis. *Revista de Saúde Pública*, *50*, 1–9.

OpenCage. (2018). *OpenCage address formatting*. Retrieved from https://github.com/OpenCageData/address-formatting

OpenStreetMap. (2018). *Osmosis*. Retrieved from https://github.com/openstreetmap/osmosis

PAF. (2018). *Postcode Address File: UK address database*. Retrieved from https://www.royalmail.com/business/services/marketing/data-optimisation/paf

Řehuřek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta, Malta: ELRA.

Reynolds, P., Behren, J. V., Gunier, R., Goldberg, D., Hertz, A., & Smith, D. (2003). Childhood cancer incidence rates and hazardous air pollutants in California: An exploratory analysis. *Environmental Health Perspectives*, *111*, 663–668.

Rubenstein, H., & Goodenough, J. B. (1965). Contextual correlates of synonymy. *Communications of the ACM*, *8*, 627–633.

Singleton, A. (2015). *Data challenges and rateable value adjustment*. Retrieved from https://www.cdrc.ac.uk/case-study/data-challenges-and-rateable-value-adjustment/

van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*, 2579–2605.

Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, *13*, 260–269.

VOA. (2017). *VOA rating list*. Retrieved from https://voaratinglists.blob.core.windows.net/html/rlidata.htm

Yancey, W. E. (2005). *Evaluating string comparator performance for record linkage* (Research Report Series, Statistics #2005-05). Washington, DC: U.S. Bureau of the Census.

Zellig, H. (1954). Distributional structure. *Word*, *10*, 146–162.

How to cite this article: Comber S, Arribas-Bel D. Machine learning innovations in address matching: A practical comparison of word2vec and CRFs. *Transactions in GIS*. 2019;23:334–348. https://doi.org/10.1111/tgis.12522