SoftwareX 17 (2022) 100965

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

rethnicity: An R package for predicting ethnicity from names

Fangzhou Xie

Department of Economics, Rutgers University, New Jersey Hall, Room 202, 75 Hamilton Street, New Brunswick, NJ 08901, United States of America

ARTICLE INFO

Article history:
Received 13 October 2021
Received in revised form 14 December 2021
Accepted 15 December 2021

Keywords:
R
LSTM
Ethnicity prediction

ABSTRACT

In this study, a new R package, `rethnicity` [1] is provided for predicting ethnicity based on names. The Bidirectional Long Short-Term Memory (Bi-LSTM), a recurrent neural network architecture commonly used for natural language processing, was chosen as the model for our study. The Florida Voter Registration was used as the training and testing data. Special care was given for the accuracy of minority groups by adjusting the imbalance in the dataset. The models were trained and exported to C++ and then integrated with R using Rcpp. Additionally, the availability, accuracy, and performance of the package were compared with other solutions.

Code metadata

Current code version	0.2.2
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-21-00198
Code Ocean compute capsule	
Legal Code License	MIT License
Code versioning system used	git
Software code languages, tools, and services used	R ($\geq$ 3.4.0)
Compilation requirements, operating environments & dependencies	Rcpp, RcppEigen, RcppThread, cli
If available Link to developer documentation/manual	https://fangzhou-xie.github.io/rethnicity/index.html
Support email for questions	https://github.com/fangzhou-xie/rethnicity/issues

1. Motivation and significance

The study on the differential effects of ethnicity requires researchers to have ethnic information available in a dataset. However, such information is usually not readily available.[2] When only names are available in the dataset, one naturally wants to predict people's ethnicity based on their names, as names are usually highly correlated with their races.

Surname analysis has been used for many years to identify ethnicity,[3] but the application of deep learning can make it even simpler, as illustrated by [2]. Moreover, first names could also be incorporated to make the prediction more precise, as given names are also correlated with ethnicity [3].

Herein, a novel approach to predict ethnicity from names is proposed and an R package is provided `rethnicity`.[4,5] The developed method achieves good performance, is fast and accessible.

2. Software description

2.1. Methodology

2.1.1. Florida voter registration dataset and undersampling

Most classification algorithms assume a relatively balanced dataset and equal misclassification cost [5]. When applying them to imbalanced data, where the instances of some classes are significantly larger or smaller than other classes, the algorithm will mainly focus on the majority class and hence ignore the

E-mail address: fangzhou.xie@rutgers.edu.

[1] https://github.com/fangzhou-xie/rethnicity. It has also been published on [CRAN].

[2] Health care is one of the areas that must be studied for ethnic disparities in insurance plans, and researchers in this field should deal with the missing ethnic information [1].

[3] See [1] for a survey.

[4] https://github.com/fangzhou-xie/rethnicity.

[5] See [4] as well. However, it should be noted that the prediction cannot be made 100% by this package. Anyone using this package should be cautious about the results and their interpretations.

https://doi.org/10.1016/j.softx.2021.100965

minority classes. One example is fraud detection, where most of the transactions are normal, but a few are fraudulent [6].

Undersampling on the majority classes will transform the imbalanced data into a balanced one. Early works [7,8] note that there are accuracy gains by undersampling the majority classes, but naive over-sampling techniques (namely resampling with replacement) do not increase accuracy. The celebrated SMOTE algorithms [9,10] for oversampling, however, work in the "feature space" and rely on the KNN algorithm, which would be infeasible to the high dimensional NLP problems. Hence plain undersampling was chosen for our classification problem. Moreover, downsizing the major classes also helps reduce the training and testing time, given that we have a massive dataset available.

The dataset from Florida Voter Registration [11] was used in this study, and it contains names and identified ethnic groups for all Florida Voters. It was extracted from Florida Voter Registration System with officially registered Florida voters as of 2017, except those who requested exemptions from public disclosure. The Florida Voter Registration dataset[6] includes nine categories of ethnicity: American Indian or Alaskan Native, Asian or Pacific Islander, Non-Hispanic Black, Hispanic, Non-Hispanic White, Other, Multi-racial, and Unknown. We restrict our attention to the four major races in the United States: Asian, Black, Hispanic, and White for the same reason as in [2] since there are not enough data points for the other categories. Our undersampling approach will take the size of the smallest group and randomly select the same number of entries for all other categories. Hence including any of the other five groups would give us an extremely small dataset and not be able for the model to learn anything from it.

2.1.2. Character-level dictionary

Classic natural language processing (NLP) models consider "tokens" the building blocks of languages. This assumption appears natural to humans because words and phrases are considered the atom of language in our daily use. However, for algorithms to process sentences, there is a need to tokenize them, create a vocabulary, and then build a model based on the vocabulary. This process will become cumbersome as the size of the data increases.[7]

Efforts to overcome this have been taken in the machine learning field to build models directly on characters instead of tokens [13,14]. It is easier to enumerate all possible characters and maintain a dictionary of those characters than a dictionary consisting of distinct tokens.[8] This way, we can keep the dictionary small[9] and avoid the out-of-vocabulary (OOV) problem.

2.1.3. Bidirectional LSTM

Long short-term memory [16, LSTM] has been widely used in sequence modeling since its proposal.[10] Moreover, [18] proposed bidirectional LSTM (Bi-LSTM), which captures the context even better than the unidirectional LSTM model.

In this package, Bi-LSTM is used as the model architecture for predicting race from names. The model was built with 256 units of an embedding layer and four Bi-LSTM layers with 512 units each. The final output layer is a dense layer of four units (equal to the number of races for the classification problem) with softmax activation function.[11,12]

The performance of the model in terms of accuracy is listed in Table 2a.

2.1.4. Distillation of knowledge

Although enjoying many advantages, training models with a character-level dictionary might be more complicated than token-level ones. Hence, large Bi-LSTM models with many parameters are trained for better accuracy. However, this trained model is enormous and would be challenging to deploy in production. Therefore, "model distillation" is used to compress the information into a smaller model.

[19] proposed the "distillation" technique for extracting information from large models and teaching a smaller model to achieve a similar prediction. To be more precise, the "student" model is trained to match the "teacher" model and the knowledge is transferred from the teacher to the student. In this way, the student will "learn" the interclass relationship better than directly learning from the data.

The distillation trick is applied on the trained large models to obtain smaller models with the same architecture but fewer parameters and layers.[13] The smaller model is compressed from the larger model and becomes the model used for inference in production.

2.1.5. Export to C++

After training student models, they are exported to C++ via `frugally-deep`[14] project. Hence, the model is no longer dependent on the installation of `Keras` (or `TensorFlow`). Subsequently, the model is loaded directly in C++ with very few dependencies.[15]

In order to make the model callable from R, an interface must be provided using Rcpp [20]. This will provide a wrapper around the underlying C++ code for loading the model and predicting the names. Additionally, the prediction can be parallelly processed by multi-threading. These features will enable the names to be processed rapidly for the prediction of ethnicity.

2.2. Comparison with other solutions

This section compares the availability, accuracy, and performance with some other packages and services. The most important comparison will be between `rethnicity` and `ethnicolr`. They have some similarities but also differ in a number of ways: first, `rethnicity` adjusts the imbalance in the dataset while `ethnicolr` does not; second, `rethnicity` uses character-level dictionary but `ethnicolr` uses bichar-level one; third, `rethnicity` considers Bi-LSTM to better capture the context between characters whereas `ethnicolr` leverages Unidirectional

[6] Further, [2] and the comment on Github (https://github.com/appeler/ethnicolr/issues/39#issuecomment-817953484) also note that it might be a limitation since the models are trained only on Florida voters. However, current evidence shows that the within-state variation is larger than the across-state one, and the model should generalize well to other states. Again, users should interpret the results cautiously.

[7] There is a need to retain a giant vocabulary, where some tokens are very common, while several are extremely infrequent [12].

[8] In the case of English, only 26 letters are needed in addition to symbols when necessary. A larger dictionary could be used by including upper-case letters. However, it is more efficient to use lower-case letters for classifying names, as upper-case letters will be fewer, and the model may not have enough opportunities to learn from the upper-case letters.

[9] Token-level models usually need a large proportion of parameters are needed to capture the vocabularies [15]. Keeping the vocabulary small also increases efficiency and makes the models focus more on the characters' relationship, capturing the context better.

[10] Some of the most recent and exciting developments in LSTM include BERT [17] and its variants.

[11] The test accuracy is given in Section 2.2.2.

[12] Last name and full name models have the same architecture but differ in the dataset used for training.

[13] The architecture of the student model includes Embedding, Bi-LSTM, and dense layers. However, there are only 32 units for Embedding, 64 units each for the two layers of Bi-LSTM.

[14] https://github.com/Dobiasd/frugally-deep.

[15] The `frugally-deep` is a lightweight header-only C++ project that depends only on FunctionalPlus, Eigen, and Json projects, which are all header-only projects. The dependency on Eigen is replaced by RcppEigen later.

Table 1
Comparison across some publicly available services/packages for predicting ethnicity from names. `rethnicity` provides a free and light-weight package for the R community without rate-limiting.

	Ethnicity	Ethnicolr	NamePrism	nationalize.io
Cost	free	free	free	paid
Rate Limit	No	No	Yes	Yes
Dependency	Low	High	N/A	N/A
Language	R	Python	API	API

LSTM. Moreover, `rethnicity` was also distilled to have a smaller model for production and exported to C++ in order to be callable from R.

2.2.1. Availability

Table 1 shows the differences between `rethnicity` and other solutions for predicting ethnicity from names. The comparison is made on four aspects: cost, rate limit, dependencies, and language.

`NamePrism` is free but is rate-limited to 60 requests per minute. `nationalize.io` offers 1000 free requests each day and requires a subscription to their services for more names to be processed in a day. `ethnicolr` might be the most similar in scope as `rethnicity`, However, it is written in Python and requires the installation of TensorFlow to run the inference.

2.2.2. Accuracy

Tables 2a and 2b present the prediction accuracy on the test data not included during the training process. Table 2a shows the accuracy of the trained teacher model, and Table 2b shows the accuracy of the student model. Note that the full name model performs better than the one that only leverages last name information for both teacher and student models. Additionally, the precision of the student model degrades compared to the teacher model but is still sufficiently high and close to that of the teacher.

Moreover, if we compare the results within each ethnic group, the accuracy for each group is roughly balanced, and the performance is slightly better for the minority groups. This suggests that the undersampling approach used in this study to adjust the imbalance (described in Section 2.1.1) in the dataset works well. If the results are compared to `ethnicolr` [2], `rethnicity` shows significantly better results on the prediction of Asian, Hispanic, and Black people, albeit the precision is lower for white people.[16]

2.2.3. Performance

The performance of the package is guaranteed by leveraging distillation for model distillation, C++, and low overhead multi-threading, as discussed in Section 2.1. However, considering speed, there is a need to test the performance rigorously and compare with it the `ethnicolr` package as a baseline.

Fig. 1 shows that the single-threaded performance is on par with that of `ethnicolr`, and the multi-threaded mode achieves further speedups. First, the distillation method successfully compresses the model and improves performance by having a smaller model. The inference speed of the single-threaded distilled model in `rethnicity` is roughly comparable to that of the larger model in `ethnicolr`. This suggests that the distillation closes the gap between the speedup led by multi-threading TensorFlow and GPU acceleration.[17] Second, there is extremely little overhead for multi-threading, and the speedup is almost linear in terms of the number of threads being used. This is clearly the feature inherited from the efficiency of the `RcppThread` package. In practice, more threads should be used to process a large dataset, depending on the size of the dataset and the total number of threads available in the machine.

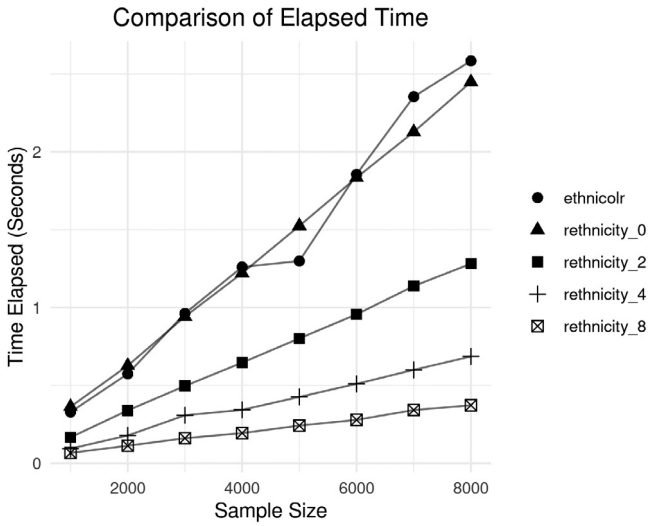

Fig. 1. Comparison of elapsed time between `rethnicity` and `ethnicolr`. At any given sample size, I run the inference five times and take the average elapsed time. Moreover, a comparison is also made for different numbers of threads available to `rethnicity`. The default single-threaded inference speed is shown as "rethnicity_0" in the plot. The performance for inference under two-thread, four-thread, and eight-thread pools is also illustrated.

2.3. Software functionalities and code snippets

The usage of the package is straightforward, as there is only one function recommended for most users.[18,19]

2.3.1. Functions arguments

There are only five arguments for the function `predict_ethnicity`: `firstnames`, `lastnames`, `method`, `threads`, and `na.rm`.

The `firstnames` argument accepts a vector of strings,[20] and is only required when `method = 'fullname'`.

`lastnames` also accepts a Character Vector and is needed for both `method = 'fullname'` and `method = 'lastname'`.

`method` can only be either `'fullname'` or `'lastname'` to indicate whether working only with last names or both first and last names.

`threads` can be chosen to have an integer greater than one to leverage multi-threading support for even faster data processing.[21]

[16] The accuracies in [2] are disproportionately high for white people, which might suggest that the classifier tends always to predict white to minimize loss.

[17] Multi-threading is the default behavior for TensorFlow, based on which `ethnicolr` is implemented. The experiment also leverages GPU for deep learning acceleration. `frugally-deep`, on the other hand, only uses single-thread and CPU.

[18] More examples can also be found at the GitHub repository: https://github.com/fangzhou-xie/rethnicity.

[19] As of package version 0.2.0., there are two more functions added for advanced users who wish to train their models and load them into the package instead of the models I trained and described in this paper. Interested readers should refer to a dedicated vignette for details (https://fangzhou-xie.github.io/rethnicity/articles/advanced_usage.html).

[20] Character Vector in R.

[21] Theoretically, one can choose a number to equal the number of threads in the machine. The more threads used, the more the overhead introduced in parallel processing, and the lesser the performance boost gained.

Table 2
Comparison between Fullname model and Lastname model.

	Fullname			Lastname			
	Precision	Recall	f1-score	Precision	Recall	f1-score	Support
asian	0.87	0.76	0.81	0.87	0.69	0.77	41861
black	0.74	0.77	0.76	0.65	0.80	0.72	41904
hispanic	0.86	0.87	0.86	0.84	0.85	0.85	41940
white	0.67	0.73	0.70	0.62	0.58	0.60	41707
total	0.79	0.78	0.78	0.74	0.73	0.73	167412

(a) Accuracy on the test data for the teacher model before distillation.

	Fullname			Lastname			
	Precision	Recall	f1-score	Precision	Recall	f1-score	Support
asian	0.86	0.73	0.79	0.84	0.64	0.73	41861
black	0.70	0.76	0.73	0.61	0.75	0.67	41904
hispanic	0.83	0.87	0.85	0.80	0.84	0.82	41940
white	0.67	0.68	0.68	0.57	0.53	0.55	41707
total	0.77	0.76	0.76	0.70	0.69	0.69	167412

(b) Accuracy on the test data for the student model after distillation.

Table 3
Comparison of total donations grouped by predicted race from donors' names. The right half of the table is taken from [2]. It should be noted, however, that we cannot compare the accuracy for the predictions made in this table, as the DIME dataset [21] does not include the ethnicity of the donors. The rigorous testing and comparison are in Section 2.2.2 and `rethnicity` shows better performance in the prediction on minority classes.

	rethnicity		ethnicolr	
	2000	2010	2000	2010
asian	6.29%	5.90%	2.00%	2.28%
black	20.83%	18.00%	8.93%	7.92%
hispanic	4.01%	4.44%	3.23%	3.31%
white	68.87%	71.66%	85.84%	86.49%

Finally, there is a `na.rm` argument. This allows one to remove missing values from the input names.[22] Otherwise, an error is thrown if values are missing in the input data. This guarantees that the model has the correct input data and returns meaningful predictions.

2.3.2. Code snippets

Here, I give one example of using the package.

```
predict_ethnicity(firstnames = "Samuel", lastnames =
      "Jackson", method = "fullname")
>   firstname lastname prob_asian prob_black prob_
    hispanic prob_white  race
> 1    Samuel  Jackson 0.01741119  0.8898849
    0.006667824  0.0860361 black
```

Listing 1: Example of the `predict_ethnicity` function.

3. Illustrative examples

To illustrate the usage of the package, I apply the prediction method to the DIME dataset [21,22, Database on Ideology, Money in Politics, and Elections].

The DIME dataset offers rich information on the finance and ideology of political campaigns. All the donors in the dataset are considered, and their races are predicted using the full-name model. The total donation amount separated by the predicted race is aggregated, and finally, the ratio of donations across ethnicity is calculated. The results for 2000 and 2010 are listed in Table 3.

Table 3 shows that, `rethnicity` suggests higher ratios of political donation when compared with `ethnicolr` results. This agrees with the accuracies in Section 2.2.2 and the discussion on the imbalanced classification problem discussed in Section 2.1.1, where `rethnicity` reduces the error for minority groups significantly. Without the adjustment, the prediction of white people will be disproportionately higher than that of minority groups, which underestimates the monetary contribution of minority groups for the elections.

4. Impact

This `rethnicity` package offers a method to infer ethnicity from names, which is usually needed for economics and political science research on the differential effects and racial discrimination.

The objective of building this package was to make the installation and usage easier for any user interested in predicting ethnicity from names for their research. The package is entirely native in R, with only dependencies being several mature packages published on CRAN. Additionally, it achieves high performance by delegating heavy computation to C++ with multi-threading. The advantages mentioned above are leveraged in the `rethnicity` package, which is free, fast, and available to the R community.

The package also adjusts balance in the training data and results in better accuracy for the ethnic minorities. The prediction error is hence balanced across ethnic groups, which would help reduce bias for further analysis.

5. Conclusions

This study demonstrates the methodology and potential usage of the `rethnicity` package in R.

It leverages different techniques to predict ethnic groups. First, undersampling was used to adjust the imbalance in the racial distribution in the dataset. Second, a character dictionary was used to reduce the dictionary size and make it independent of training data. Third, Bi-LSTM was chosen as the architecture owing to its superior performance in capturing context. Fourth, after training the gigantic teacher model, the information was distilled by instructing a much smaller student model. Finally, the student model was exported to C++ and then loaded via Rcpp.

[22] For the last name model, only non-missing names are retained for processing and are returned. The full name model will remove names if either first name or last name is missing and only process the names where both are present.

The model was trained using the Florida Voter Registration dataset using the voters' names and their identified ethnicity. After training the large model, a smaller student model was also trained and tested. Results for the testing of accuracy and performance are also provided and show its competitiveness.[23]

The code snippet is provided as an example of how to use the package. Application to finance and ideology data of political candidates is also illustrated.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Fiscella K, Fremont AM. Use of geocoding and surname analysis to estimate race and ethnicity. Health Serv Res 2006;41(4p1):1482–500. http://dx.doi.org/10.1111/j.1475-6773.2006.00551.x.

[2] Sood G, Laohaprapanon S. Predicting race and ethnicity from the sequence of characters in a name. arXiv:1805.02109.

[3] Fryer Jr RG, Levitt SD. The causes and consequences of distinctively black names. Q J Econ 2004;119(4):767–805. http://dx.doi.org/10.1162/0033553041502180.

[4] Xie F. Rethnicity: predicting ethnicity from names. arXiv:2109.09228.

[5] Sun Y, Wong AKC, Kamel MS. Classification of imbalanced data: A review. Int J Pattern Recogn Artif Intell 2009;23(04):687–719. http://dx.doi.org/10.1142/S0218001409007326.

[6] Fawcett T, Provost F. Adaptive fraud detection. Data Min Knowl Discov 1997;1(3):291–316. http://dx.doi.org/10.1023/A:1009700419189.

[7] Ling CX, Li C. Data mining for direct marketing: Problems and solutions. In: Proceedings of the fourth international conference on know ledge discovery and data mining. New York: AAAI Press; 1998, p. 7.

[8] Japkowicz N. The class imbalance problem: Significance and strategies. In: In proceedings of the 2000 international conference on artificial intelligence. 2000, p. 111–7.

[9] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: Synthetic minority over-sampling technique. J Artif Intell Res 2002;16:321–57. http://dx.doi.org/10.1613/jair.953.

[10] Fernandez A, Garcia S, Herrera F, Chawla NV. SMote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. J Artif Intell Res 2018;61:863–905. http://dx.doi.org/10.1613/jair.1.11192.

[11] Sood G. Florida voter registration data. 2017, http://dx.doi.org/10.7910/DVN/UBIG3F.

[12] Zipf GK. The psycho-biology of language: An introduction to dynamic philology. George Routledge & Sons; 1936, first printing edition Edition.

[13] Zhang X, Zhao J, LeCun Y. Character-level convolutional networks for text classification. In: Advances in neural information processing systems, Vol. 28. Curran Associates, Inc.; 2015.

[14] Sutskever I, Martens J, Hinton G. Generating text with recurrent neural networks. In: Proceedings of the 28th international conference on international conference on machine learning. Madison, WI, USA: Omni Press; 2011, p. 1017–24.

[15] Xue L, Barua A, Constant N, Al-Rfou R, Narang S, et al. ByT5: towards a token-free future with pre-trained byte-to-byte models. arXiv:2105.13626.

[16] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9(8):1735–80. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[17] Devlin J, Chang M-W, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805.

[18] Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw 2005;18(5):602–10. http://dx.doi.org/10.1016/j.neunet.2005.06.042.

[19] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv:1503.02531.

[20] Eddelbuettel D, Francois R. Rcpp: Seamless R and C++ integration. J Statist Softw 2011;40(1):1–18. http://dx.doi.org/10.18637/jss.v040.i08.

[21] Bonica A. Database on ideology. In: Money in politics, and elections. 2019, http://dx.doi.org/10.7910/DVN/O5PX0B.

[22] Bonica A. Mapping the ideological marketplace. Am J Polit Sci 2014;58(2):367–86. http://dx.doi.org/10.1111/ajps.12062.

[23] However, the predictions being made by the package are not 100% correct, and researchers should be cautious when interpreting the results. Moreover, the predicted ethnicity information should be considered a covariate being measured with errors, which may bias the regression estimates.