Appendix

It's All in the Name: A Character Based Approach to Infer Religion

Rochana Chaturvedi
University of Illinois, Chicago

Sugat Chaturvedi
University of Sussex

Table of Contents

A Inferring Race/Ethnicity in the US

We extend our approach beyond South Asia to the problem of race/ethnicity inference from names in the North American context. For this, we use the publicly available voter registration data from North Carolina comprising over 8 million registered voters.[1] We classify names into five categories—Hispanics, and non-Hispanic Whites, Blacks, Asians, and Others.[2,3]

We use the Bayesian improved surname geocoding (BISG) implementation given by Clark, Curiel, and Steelman (2021) in their R package *zipWRUext* as our baseline. It combines ZIP codes and last names using Bayes' rule to impute ethnicity.[4] This approach is shown to have higher coverage and is a more accessible alternative than the more expensive geocoding approach. For character-based models, we pre-process the names by upper-casing and retaining only white-spaces and alphabetical characters. We use first name, middle name, and surname. In addition, we incorporate prior knowledge of racial/ethnic composition within a zipcode using the 2018 American Community Survey.[5] We train language model, Logistic Regression, SVM, and CNN classifiers.

Table 5. Evaluation results for North Carolina voters Test Set. The table presents the Precision (P), Recall (R), and their harmonic mean (F_1 score) as well as coverage for all the models. Standard errors are reported in parentheses. The observations considered for BISG only include those names that could be classified unambiguously by this method.

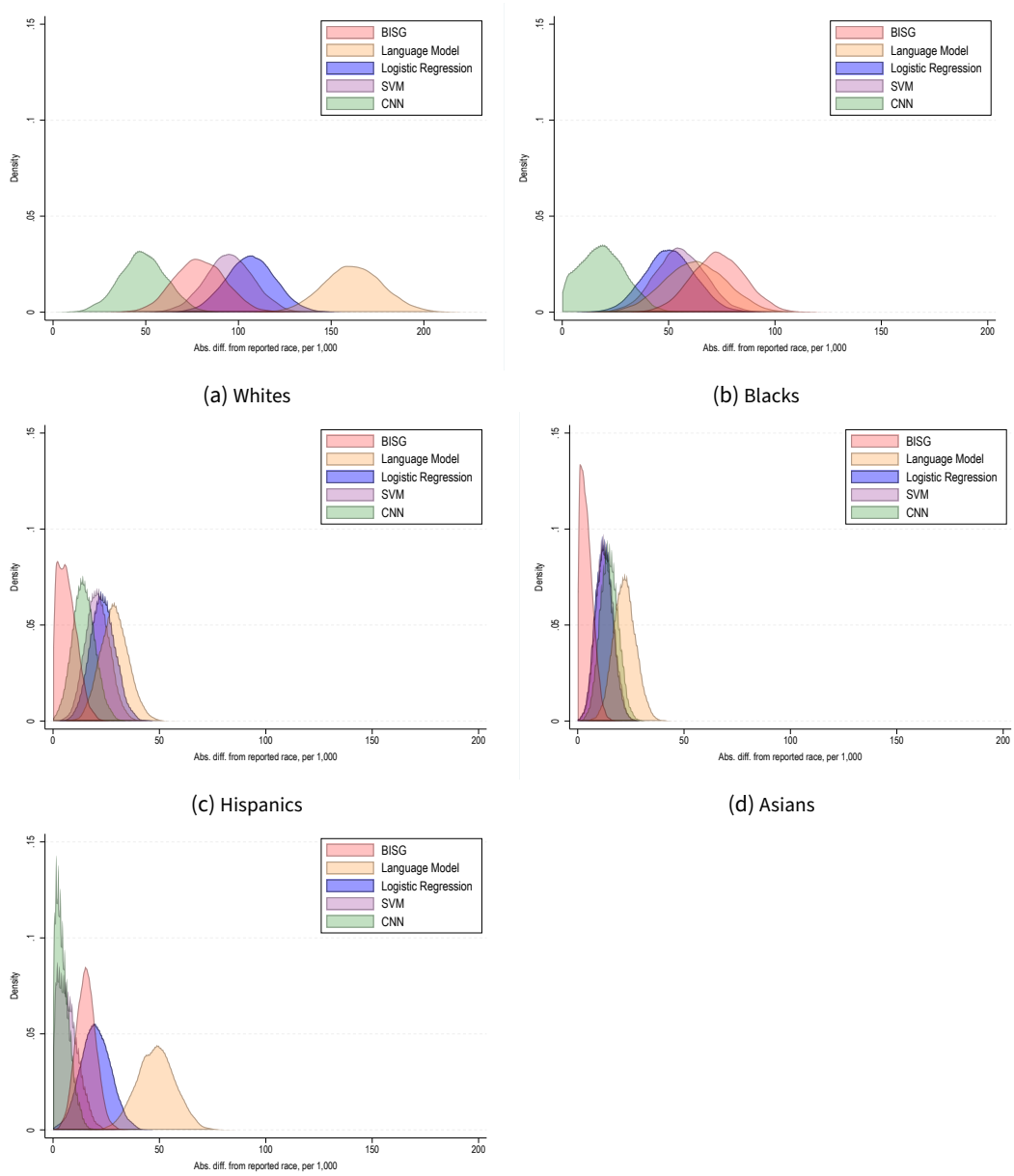
Models	Coverage	F_1	White		Black		Hispanic		Asian		Others	
			P	R	P	R	P	R	P	R	P	R
BISG	86.06	60.89	83.20	**93.19**	**70.54**	45.29	**71.89**	76.49	**52.51**	64.36	**61.65**	19.33
			(0.11)	(0.10)	(0.27)	(0.18)	(0.44)	(0.39)	(0.76)	(0.71)	(1.11)	(0.53)
Language Model	99.64	48.17	88.24	68.18	46.33	60.51	47.24	79.16	27.18	74.24	8.25	25.47
			(0.12)	(0.13)	(0.17)	(0.24)	(0.34)	(0.52)	(0.48)	(0.95)	(0.34)	(0.71)
LR	100.00	59.13	**93.01**	79.13	61.08	76.07	54.01	83.61	39.54	77.98	18.67	**34.78**
			(0.11)	(0.11)	(0.16)	(0.21)	(0.32)	(0.46)	(0.52)	(0.83)	(0.40)	(0.62)
SVM	100.00	60.71	92.91	80.54	60.85	**77.20**	56.38	**84.01**	40.04	80.02	25.47	31.96
			(0.11)	(0.11)	(0.16)	(0.20)	(0.33)	(0.45)	(0.52)	(0.81)	(0.49)	(0.60)
CNN	100.00	**62.02**	91.50	85.28	66.42	72.38	60.94	81.69	38.35	**82.60**	29.93	30.81
			(0.10)	(0.10)	(0.17)	(0.19)	(0.34)	(0.42)	(0.49)	(0.77)	(0.53)	(0.57)
Observations	189,615		135,533		39,134		8,095		2,448		4,405	

We report the results in Table 5. First, we find that BISG has a relatively low coverage at around 86% while the character-based models are able to classify 100% of individuals. Secondly, BISG classifies individuals overwhelmingly into the majority White group as indicated by a high recall but low precision for Whites. This is particularly problematic for the Blacks who are the largest minority group in the US. The recall for Blacks is only 45% using BISG. In other words, less than half of true Blacks are actually predicted as Blacks indicating that BISG systematically undercounts them. On the other hand for CNN, which is our best performing model overall (with a macro-average F_1 score of 62%), the recall for Blacks is over 70%.

This becomes even more apparent in Figure 4 which shows the absolute difference between actual and estimated race counts on 10,000 bootstrap samples with a sample size of 1,000 per draw. BISG performs the worst in estimating the Black population share. The difference in median accuracy between CNN and BISG is 54.27 per 1,000 for Blacks. CNN also aggregates better than

1. The data are available at https://www.ncsbe.gov/results-data/voter-registration-data.
2. In line with prior work, we combine Native Hawaiians and Pacific Islanders with Asians while we combine American Indians and Alaska Natives with Others (Imai and Khanna 2016). The population share of Whites is 71.48%, Blacks is 20.64%, Hispanics is 4.27%, Asians is 1.29%, and Others comprise 2.32% in our data.
3. Since we have a large sample size, we split it into training, validation, and test sets in the ratio 94:3:3. We undersample the training set so that all the classes have the same number of observations in the training set. This reduces our final training data to 383,485 observations.
4. The implementation is available at https://github.com/jcuriel-unc/zipWRUext. We use the 2018 American Community Survey (ACS) to obtain group compositions within each ZIP code.
5. In case of missing zip codes, we use the state-wide group composition in our data as the default composition.

BISG for the Whites and Others group with the median difference being 30.02 and 11.42 per 1,000 respectively. On the other hand, BISG estimates Hispanic and Asian counts better than CNN with median difference in accuracy of 7.82 and 11.17 (per 1,000) respectively.



(a) Whites

(b) Blacks

(c) Hispanics

(d) Asians

(e) Others

Figure 4. The figure shows the density plot of the absolute difference between reported and estimated race counts per 1,000 people based on the North Carolina voters test set.

In general, the performance for race classification in the US is worse than religion classification in South Asia. This is consistent with our discussion in Section 2. Owing to a lack of clear linguistic theory distinguishing White and Black names and assimilation of various racial/ethnic groups, the relatively lower performance is unsurprising.[6] Moreover, the "Others" class includes a mix of White, Black, Hispanic, and Asian sounding names. This is further validated in Table 6 which shows confusion matrices for BISG and CNN models on the test set. We see that "Others" are variously classified among Whites, Blacks, Hispanics, and Asians. In conclusion, compared to BISG, CNN provides 100% coverage and is able to remove bias in counting Black minority.

Table 6. Confusion Matrices for BISG (left) and CNN (right) models for the North Carolina voters test set. W = "Whites", B = "Blacks", H = "Hispanics", A = "Asians", O = "Others"

		Predicted					
		W	B	H	A	O	Total
True	W	107,764	5,791	1,407	444	229	115,635
	B	17,833	15,327	313	132	234	33,839
	H	1,466	200	5,727	83	11	7,487
	A	643	56	80	1,421	8	2,208
	O	1,814	355	439	626	775	4,009
	Total	129,520	21,729	7,966	2,706	1,257	163,178

		Predicted					
		W	B	H	A	O	Total
True	W	115,578	13,256	2,994	1,601	2,104	135,533
	B	8,819	28,325	562	523	905	39,134
	H	804	419	6,613	168	91	8,095
	A	187	50	112	2,022	77	2,448
	O	923	596	570	959	1,357	4,405
	Total	126,311	42,646	10,851	5,273	4,534	189,615

B Model Details

Name2community Name2community (Susewind 2015) counts frequency of each name part within a religious class in a given reference list using spelling (S) and pronunciation (P) matches derived from the fuzzy Indic Soundex algorithm. It then computes a certainty index I for each name part X for each community Y using the formula given below and multiplies it by "quality factors" based on spelling and pronunciation q_S and q_P defined as the percentage of unambiguous name parts in the reference list:

$$\frac{I(X \in Y)}{q_S \cdot q_P} = \left(1 - \frac{S_X - S_{X,Y}}{S_X} \times \frac{P_X - P_{X,Y}}{P_X}\right)$$

These indices are then aggregated over all name parts to get the certainty index for the entire name N belonging to a certain community as follows:

$$I(N \in Y) = 1 - \left(\prod_X \frac{E_X - I(X \in Y)}{E_X}\right)$$

Where, E_X is the total number of matches for X in the reference list.

Language Model This approach is based on training multiple probabilistic n-gram language models—one for each class. A standard metric to evaluate language models is perplexity. It measures how well the probability distribution learned by a model represents an example in the test set. Mathematically, perplexity is the inverse probability of a character n-gram sequence $c_1 c_2 \ldots c_n$, normalized by its length:

$$Perplexity = \sqrt[n]{\frac{1}{P(c_1 c_2 \ldots c_n)}}$$

For classifying the test set names, we compute the perplexities of the given name for the language model corresponding to each class and assign it the class having the lower perplexity

6. There is also a possibility of some misreporting of ethnicity/race. For example, "Pang Yeng Chang" and "Sanjay Bhulabhai Patel" are labeled as Hispanic in the test set but predicted as Asian by the CNN model while "Miriam Gonzalez" is labeled as Asian but predicted as Hispanic by the CNN model.

score. We use NLTK package for running our experiments and experiment with various n-gram ranges and different smoothing and interpolation techniques to address the problem of pattern sparsity in the training corpus.

TF-IDF For each token t and document d, the TF-IDF score is calculated as follows in our implementation:

$$TF - IDF(t, d) = TF(t, d) \times IDF(t)$$

$$TF(t, d) = \frac{N_{t,d}}{N_d} \quad \text{and,}$$

$$IDF(t) = ln\frac{1 + n}{1 + DF(t)} + 1$$

Where $N_{t,d}$ is number of occurrences of token t in document d; N_d is document length; $DF(t)$ is the number of documents containing token t; and n is the count of documents in the corpus.

CNN After zero padding each name, each of its character is converted to an embedding. Thereafter it is passed through a 1-D convolution with max pooling over time. The name representation so obtained is finally passed through a fully connected layer with softmax activation to obtain probabilities over the output classes. The model is trained to minimize binary cross-entropy loss with balanced class weights. The Kernel weights are randomly initialized and dropout is used to prevent overfitting. We use Nadam optimizer (Dozat 2016) using mini-batches for 80 epochs and reduce learning rate if our validation loss does not improve. The best performing model on REDS validation dataset is selected.

Additional Models We also experiment with Long short-term memory (LSTM) and CNN-LSTM architectures (results are available on request). We combine the probabilities for an individual and their parent/spouse using a two-stage model as well. These models are described below:

- **LSTM** Recurrent Neural Networks (RNN) are designed to learn from sequential input (Elman 1990). LSTM is an RNN variant to handle long range dependencies by allowing the network to learn adaptively via gating mechanism (Hochreiter and Schmidhuber 1997). Therefore, LSTM is widely used for NLP tasks. In our implementation, we apply a linear transformation to each hidden state output of the LSTM layer and apply max-pooling over transformed sequences to get the name encoding. The encoding is further transformed via a mapping inspired by highway layer (Srivastava, Greff, and Schmidhuber 2015), wherein we use a linear transformation followed by ELU activation and concatenate the transform and carry gate outputs.
- **CNN-LSTM** Inspired by Kim *et al.* (2016), our third neural network architecture combines CNN with LSTM. The embedding outputs are fed to separate CNN-LSTM stacks. Each such stack has a fixed CNN kernel width k, varying from 2–6 and multiple filters. We use $k - 1$ max pooling on CNN output. We then feed this sequence of most relevant k-grams encodings to an LSTM layer. We also train another LSTM layer directly from input sequence. For each LSTM layer, we perform a linear transformation over its output sequence and apply global max pooling to get k-gram name representation. Finally, we concatenate all six representations and pass them through a fully connected layer. We report the range of hyperparameter search and the hyperparameter choice in Tables 8 and 9 respectively.
- **Two-Stage Models** Here we combine probabilities P_1 and P_2 (or confidence scores in case of SVM) for name of the person and that of their parent/spouse respectively and handcrafted features

derived from them using a linear SVM model with l2 regularization. The final confidence score C_M is then represented as:

$$C_M = F(f_k(P_1), f_k(P_2), g_k(P_1, P_2))$$

Where, f_k and g_k denote handcrafted features:

$$f_k \in \{P_i, log(P_i)\}; \quad g_k \in \{P_1 \cdot P_2, max(P_1, P_2), P_1 \cdot log(P_2), P_2 \cdot log(P_1), max(log(P_1), log(P_2))\}$$

We take these features to implement a non-linear decision boundary separating Muslim and non-Muslim names based on P_1 and P_2. For the final results, we use recursive feature elimination (RFE) to select only the relevant features from our feature pool. For a given feature count, RFE chooses the set of features contributing the most to the predictive power of the model. We then choose the *optimal* number of features as those that have the highest macro-average recall on the validation set.

Table 7. Evaluation Results on both REDS and U.P. Rural Households test sets. The table presents the Precision (P), Recall (R), and their harmonic mean (F_1 score) for the two-stage models. Standard errors are reported in parentheses.

		REDS					U.P. Rural Households				
	Models	F_1	**Muslim**		**Non-Muslim**		F_1	**Muslim**		**Non-Muslim**	
			P	**R**	**P**	**R**		**P**	**R**	**P**	**R**
Two-Stage Models	Logistic Regression	96.87	92.42	96.29	99.63	99.21	95.55	89.35	95.49	99.30	98.25
			(0.30)	(0.31)	(0.10)	(0.10)		(0.26)	(0.28)	(0.11)	(0.11)
	SVM	96.94	92.75	96.19	99.62	99.25	**97.38**	**95.15**	95.76	99.35	**99.25**
			(0.30)	(0.31)	(0.10)	(0.10)		(0.21)	(0.21)	(0.08)	(0.08)
	CNN	**97.17**	**92.97**	96.86	99.68	**99.27**	94.75	85.12	**97.75**	**99.65**	97.38
			(0.29)	(0.30)	(0.09)	(0.09)		(0.27)	(0.31)	(0.11)	(0.12)
	LSTM	96.45	90.49	96.86	99.68	98.98	93.04	80.38	97.52	99.61	96.34
			(0.32)	(0.34)	(0.10)	(0.11)		(0.30)	(0.36)	(0.13)	(0.14)
	CNN-LSTM	96.91	91.89	**97.05**	**99.70**	99.14	95.02	86.70	96.73	99.49	97.72
			(0.30)	(0.31)	(0.10)	(0.10)		(0.27)	(0.30)	(0.11)	(0.12)
	Observations	11,543	1,051		10,492		20,000	2,663		17,337	

These additional two-way classification results are shown in Table 7. Most results remain qualitatively similar. However, in panel C, which shows the results for the two-stage model, the recall for Muslim class improves substantially for the neural models. This is due to better separation between Non-Muslim and Muslim households for these models in the (P_1,P_2) space. This is an important result because of class imbalance in the data. Thus, neural models are now better able to classify actual Muslims and the models are less biased towards classifying a household as belonging to the majority non-Muslim class. This implies that the two-stage neural models can be expected to outperform the other models in areas with relatively high Muslim population share. Chaturvedi, Das, and Mahajan (2021) find that our two-stage CNN-LSTM model generalizes very well at an aggregate level using names from a census of over 25 million households in rural Uttar Pradesh. They report a correlation of 97.8% between the Muslim household share at the sub-district (tehsil) level predicted using our model and the Muslim population share reported in the 2011 census.

C Hyperparameters

Table 8. Hyperparameter Range for Binary Classifiers. This table presents the range of hyperparameters for training our binary classifiers to infer Muslim vs non-Muslim class. The final parameter choices are based on evaluation results on REDS validation set.

Parameter	Experimental Range
Embedding dimension	5-80
CNN kernel sizes (k)	1–7
CNN filters	150–300
CNN activation	ELU, ReLU, tanh
LSTM hidden units	100-750
LSTM Direction	forward, reverse, bidirectional
LSTM pooling	first, last, average, max
Embedding dropout	0–0.3
Dropout	0–0.5
Highway layers	1–3
Transformation activation	ELU, ReLU, tanh
Highway layer output	add, concatenate
Loss	binary cross-entropy, focal-loss
Batch size	32-1024
Optimizer	Adam, Nadam, rmsprop
Kernel initialization	He uniform, Glorot uniform, Glorot normal, Lecun uniform
l2 regularization parameter	0–100
TF-IDF Max n-grams	1–12

Table 9. Hyperparameter Choice for Binary Classifiers. In this table we present the final hyperparameter choices for our binary classifiers. For the LSTM only model, we reduce LSTM hidden units to 100 with 0.15 dropout rate. For CNN only model, we also use CNN kernel size of 1. All binary classification results reported on held-out test sets use these model

Parameter	Choice
Embedding dimension	29 (CNN); 30 (CNN concat)
Kernel initialization	He uniform
Embedding dropout	0.1
Dropout	0.25
CNN kernel sizes (k)	{2, 3, 4, 5, 6}
CNN filters	$\min(300, 50 \cdot k + 100)$
CNN activation	ELU
LSTM hidden units	250
Highway carry bias	-2
Transform activation	tanh
epochs	30
l2 regularization parameter	6.95 (LR), 7.50 (LR concat); 0.21 (SVM), 0.32 (SVM concat)
TF-IDF Max n-grams	5 (LR), 9 (LR concat); 9 (SVM), 10 (SVM concat)

D Analysis Based on Name Parts

To understand which part in a name contributes the most to predictions for a certain class, we perform LRP on all the correctly classified names in the test set. We then divide each observation into name parts (i.e. first name, last name etc.), and map relevance scores to normalized length of name part such that the maximum length of a name part is 1. We use absolute values of character relevance scores. Figure 5 shows local polynomial plot of relevance scores over name length. We find that for correctly classified Muslims (left panel), major part of the relevance is attributed to the end of the first name part and to the beginning of the second name part. For correctly classified Hindus (right), however, the highest relevance for the Hindu class is concentrated in the middle of each name part—especially the second name part. In both the panels, we find that the latter name parts are considered less important by our classifier.

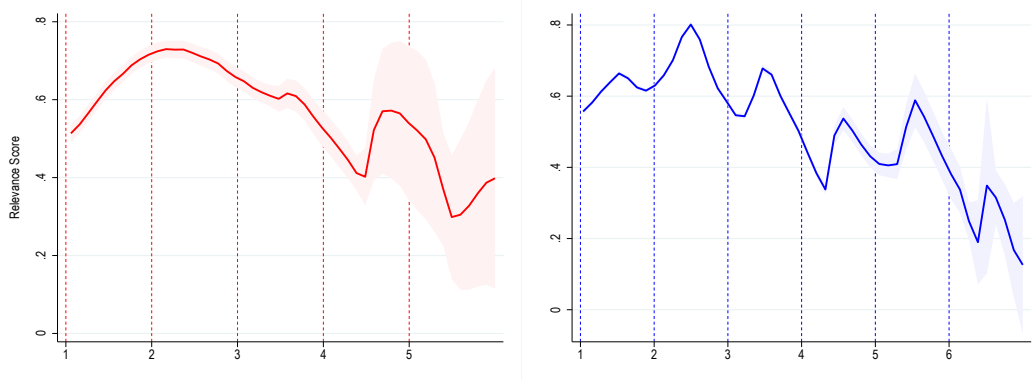
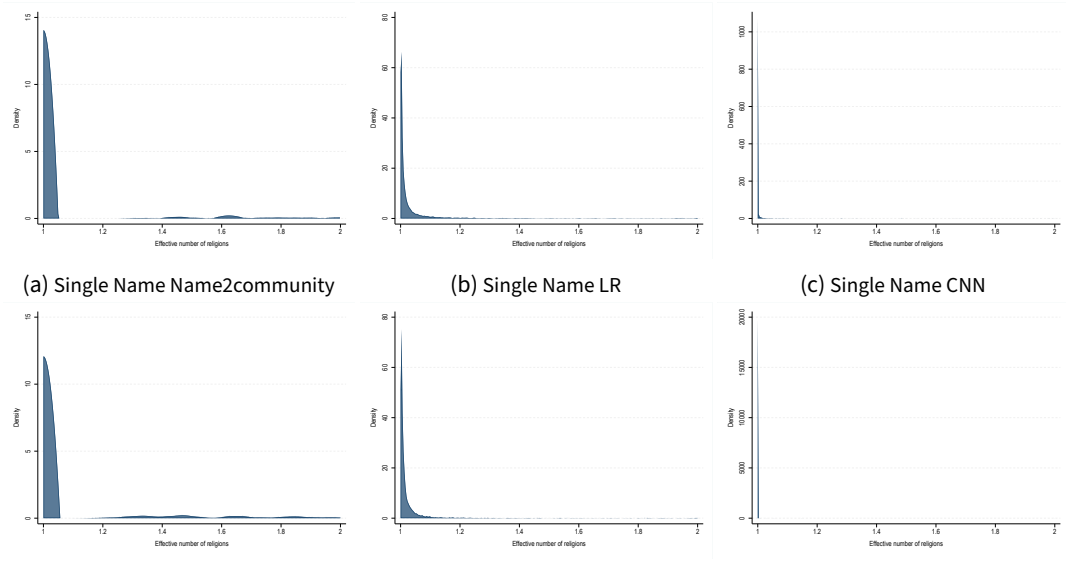


Figure 5. Character Relevance Over Name Length. Character relevance distribution over name length using LRP on REDS test sample. Left panel shows Muslim relevance for True Muslims in the sample and right panel shows Non-Muslim relevance for true Non-Muslims. The x-axis represents i^{th} name part. The shaded region denotes 95% confidence interval.

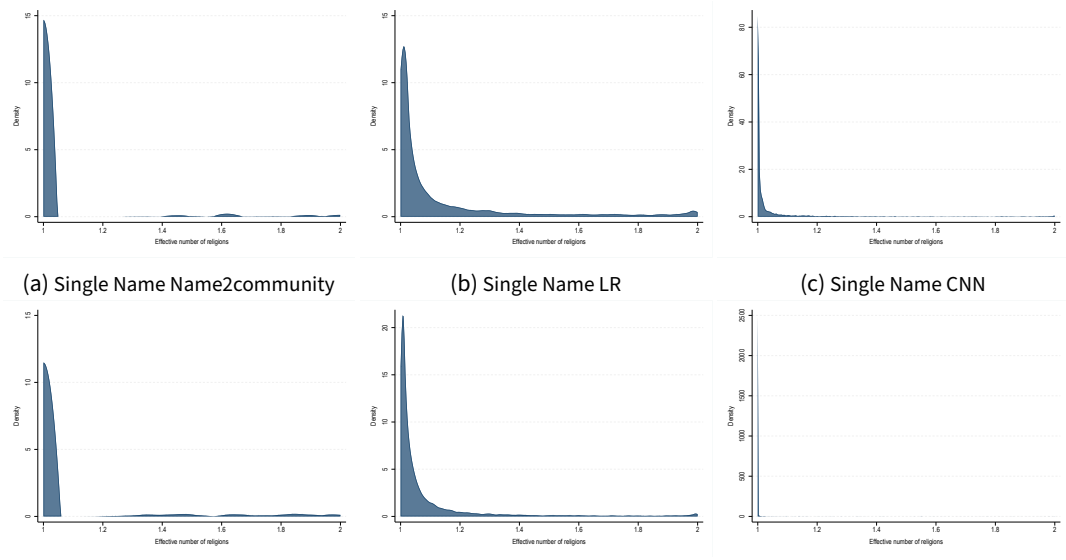
E Effective Number of Imputed Religions and Average Error Rates



(a) Single Name Name2community (b) Single Name LR (c) Single Name CNN

(d) Concatenated Name2community (e) Concatenated LR (f) Concatenated CNN

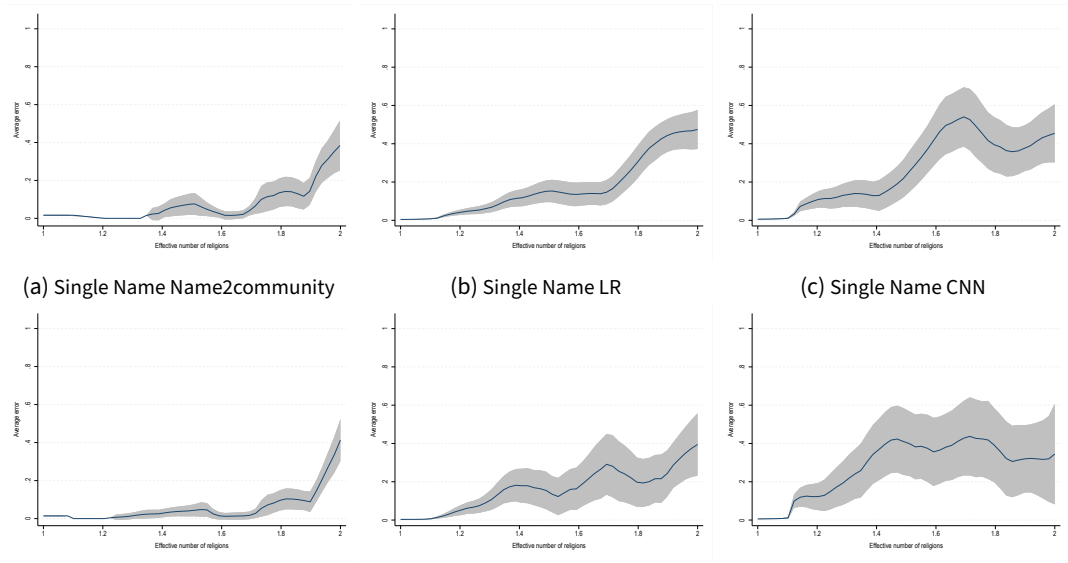
Figure 6. The figure shows the density plot for effective number of imputed religions, defined as the inverse of Herfindahl index based on model probabilities, at the individual level for the REDS test set. For Name2community, we normalize the certainty index so that the scores for Muslims and non-Muslims sum up to 1. We are unable to compute effective number of religions for SVM and the language model approach as they do not directly return probabilities.



(a) Single Name Name2community (b) Single Name LR (c) Single Name CNN

(d) Concatenated Name2community (e) Concatenated LR (f) Concatenated CNN

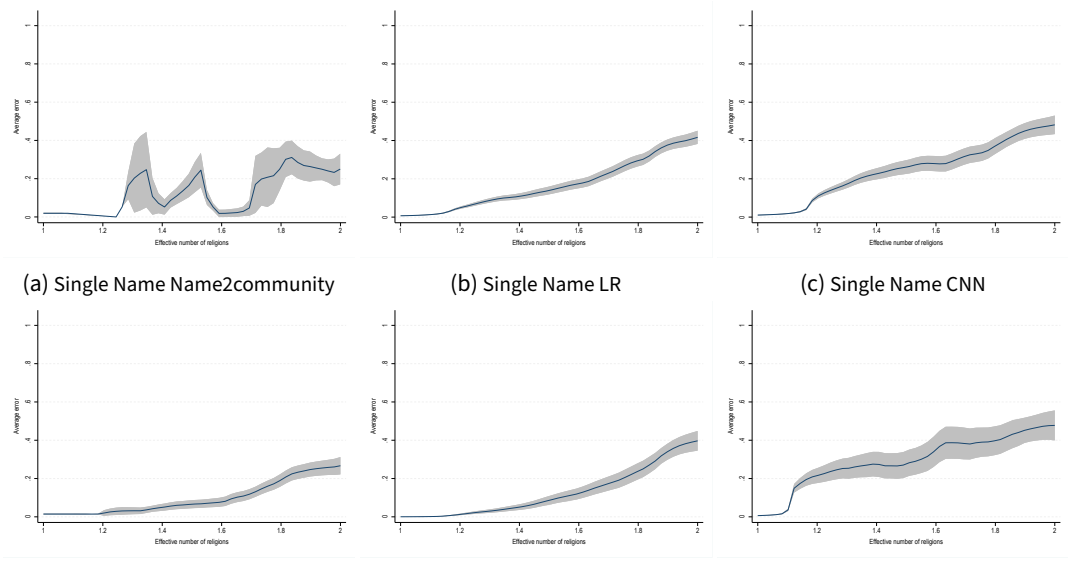
Figure 7. The figure shows the density plot for effective number of imputed religions, defined as the inverse of Herfindahl index based on model probabilities, at the individual level for the U.P. Rural Households test set. For Name2community, we normalize the certainty index so that the scores for Muslims and non-Muslims sum up to 1. We are unable to compute effective number of religions for SVM and the language model approach as they do not directly return probabilities.



(a) Single Name Name2community (b) Single Name LR (c) Single Name CNN

(d) Concatenated Name2community (e) Concatenated LR (f) Concatenated CNN

Figure 8. The figure shows the average error rates by the effective number of imputed religions, defined as the inverse of Herfindahl index based on model probabilities, at the individual level for the REDS test set. For Name2community, we normalize the certainty index so that the scores for Muslims and non-Muslims sum up to 1. We are unable to compute effective number of religions for SVM and the language model approach as they do not directly return probabilities or certainty measures.



(a) Single Name Name2community (b) Single Name LR (c) Single Name CNN

(d) Concatenated Name2community (e) Concatenated LR (f) Concatenated CNN

Figure 9. The figure shows the average error rates by the effective number of imputed religions, defined as the inverse of Herfindahl index based on model probabilities, at the individual level for the U.P. rural households test set. For Name2community, we normalize the certainty index so that the scores for Muslims and non-Muslims sum up to 1. We are unable to compute effective number of religions for SVM and the language model approach as they do not directly return probabilities or certainty measures.

F Gender

Table 10. Evaluation Results on both REDS and U.P. Rural Households test sets for women. The table presents the Precision (P), Recall (R), and their harmonic mean (F_1 score) for the baseline model Name2community as well as our character-based machine learning models for the REDS test set. Standard errors are reported in parentheses. The coverage for character-based machine learning models is 100% while both the baselines do not give full coverage. The observations considered for Name2community and Language Model baselines only include those names that could be classified unambiguously by these methods.

		REDS						U.P. Rural Households					
	Models	Coverage	F_1	Muslim		Non-Muslim		Coverage	F_1	Muslim		Non-Muslim	
				P	R	P	R			P	R	P	R
Panel A: Single Name	Name2community	57.58	80.05	67.44	59.18	96.55	97.56	58.97	**87.56**	**90.00**	66.94	**97.31**	**99.38**
				(3.28)	(2.89)	(0.89)	(0.85)			(1.82)	(1.37)	(0.45)	(0.40)
	Language Model	97.41	87.21	74.34	80.00	97.77	96.94	98.46	82.78	62.06	**81.54**	97.28	92.97
				(1.88)	(2.02)	(0.65)	(0.67)			(1.19)	(1.53)	(0.52)	(0.57)
	Logistic Regression	100.00	**93.83**	88.07	**89.72**	**98.87**	98.67	100.00	85.60	70.78	79.76	97.08	95.34
				(1.38)	(1.40)	(0.46)	(0.46)			(1.20)	(1.33)	(0.48)	(0.50)
	SVM	100.00	92.41	84.07	88.79	98.76	98.15	100.00	86.19	72.98	79.15	97.02	95.85
				(1.49)	(1.57)	(0.51)	(0.52)			(1.20)	(1.29)	(0.47)	(0.49)
	CNN	100.00	93.38	**90.20**	85.98	98.47	**98.97**	100.00	85.68	84.35	66.77	95.43	98.25
				(1.47)	(1.40)	(0.47)	(0.47)			(1.41)	(1.13)	(0.47)	(0.43)
Panel B: Concatenated	Name2community	70.15	93.27	95.31	81.33	97.99	99.56	80.12	94.31	95.19	84.98	98.19	99.48
				(1.85)	(1.59)	(0.56)	(0.53)			(0.99)	(0.89)	(0.33)	(0.31)
	Language Model	96.95	91.78	86.00	84.31	98.31	98.52	96.29	91.85	86.36	84.98	97.92	98.14
				(1.63)	(1.60)	(0.53)	(0.52)			(1.02)	(1.00)	(0.38)	(0.37)
	Logistic Regression	100.00	96.67	92.73	95.33	99.49	99.18	100.00	95.86	95.22	90.33	98.64	99.36
				(1.02)	(1.05)	(0.34)	(0.35)			(0.74)	(0.70)	(0.27)	(0.26)
	SVM	100.00	96.70	91.96	**96.26**	**99.59**	99.08	100.00	**96.78**	95.37	**93.35**	**99.06**	99.36
				(1.01)	(1.05)	(0.34)	(0.35)			(0.65)	(0.63)	(0.24)	(0.24)
	CNN	100.00	**96.81**	**97.03**	91.59	99.08	**99.69**	100.00	92.76	**99.23**	77.64	96.93	**99.91**
				(1.04)	(0.99)	(0.33)	(0.33)			(1.03)	(0.82)	(0.34)	(0.31)
	Observations	1,082		107		975		2,671		331		2,340	

G Multi-way Religion Classification

Table 11. Evaluation results for REDS Test Set. The table presents the Precision (P), Recall (R), and their harmonic mean (F_1 score) as well as coverage for the baseline Name2community and language model as well as our character-based machine learning models for the REDS test set. Standard errors are reported in parentheses. The observations considered for Name2community only include those names that could be classified unambiguously by this method.

	Models	Coverage	F_1	Buddhist		Christian		Hindu		Jain		Muslim		Sikh	
				P	R	P	R	P	R	P	R	P	R	P	R
Panel A: Single Name	Name2community	58.35	44.14	0.00	0.00	26.77	44.62	92.66	95.27	26.19	26.83	90.36	87.84	43.40	14.62
				(7.17)	(4.24)	(1.58)	(1.78)	(0.30)	(0.27)	(4.28)	(3.80)	(0.89)	(0.77)	(2.20)	(1.12)
	Language Model	97.69	47.47	11.63	35.71	18.36	68.29	96.43	82.46	5.23	32.00	80.73	87.46	43.25	81.45
				(1.81)	(4.54)	(0.69)	(1.92)	(0.24)	(0.32)	(1.51)	(5.37)	(0.66)	(0.98)	(0.83)	(1.63)
	Logistic Regression	100.00	71.60	64.29	73.97	64.04	69.36	97.95	94.94	28.92	46.15	93.17	93.82	55.23	**91.11**
				(2.28)	(2.75)	(0.98)	(1.15)	(0.18)	(0.19)	(2.29)	(3.26)	(0.53)	(0.59)	(0.69)	(1.00)
	SVM	100.00	**76.90**	**89.09**	67.12	**81.27**	63.90	97.39	**97.39**	**56.41**	42.31	**95.13**	93.25	**65.43**	83.48
				(2.65)	(2.31)	(1.08)	(0.96)	(0.16)	(0.16)	(3.15)	(2.74)	(0.50)	(0.50)	(0.74)	(0.84)
	CNN	100.00	68.08	53.33	**76.71**	52.21	**70.07**	**98.06**	93.20	23.21	**50.00**	90.49	**94.59**	51.81	90.74
				(2.15)	(3.04)	(0.93)	(1.27)	(0.19)	(0.22)	(2.08)	(3.61)	(0.54)	(0.66)	(0.71)	(1.11)
Panel B: Concatenated	Name2community	67.34	42.42	0.00	0.00	34.19	57.14	92.67	95.55	21.21	33.33	90.58	89.53	5.32	0.97
				(9.11)	(3.50)	(1.46)	(1.55)	(0.27)	(0.23)	(3.36)	(3.46)	(0.82)	(0.67)	(2.82)	(0.99)
	Language Model	97.24	53.65	28.92	34.29	19.75	67.36	96.55	87.40	13.46	26.92	86.09	88.42	51.29	81.75
				(2.76)	(4.04)	(0.70)	(1.73)	(0.22)	(0.28)	(2.47)	(4.68)	(0.65)	(0.88)	(0.86)	(1.46)
	Logistic Regression	100.00	82.76	76.00	**78.08**	77.18	**75.53**	**98.49**	97.75	75.00	57.69	95.94	**96.24**	76.11	**93.10**
				(1.95)	(2.02)	(0.83)	(0.84)	(0.14)	(0.14)	(2.68)	(2.40)	(0.43)	(0.44)	(0.65)	(0.74)
	SVM	100.00	**82.88**	**92.86**	71.23	**88.99**	69.12	97.98	**98.83**	**78.12**	48.08	**96.54**	95.99	**83.69**	85.66
				(2.19)	(1.82)	(0.91)	(0.76)	(0.14)	(0.13)	(2.90)	(2.15)	(0.41)	(0.39)	(0.69)	(0.66)
	CNN	100.00	78.94	65.82	71.23	69.62	69.12	98.14	97.44	59.26	**61.54**	95.09	96.11	76.78	88.20
				(2.05)	(2.18)	(0.89)	(0.91)	(0.15)	(0.15)	(2.48)	(2.58)	(0.46)	(0.47)	(0.72)	(0.79)
	Observations	17,207		73		421		14,540		52		1,570		551	

In this section we discuss the details of the multi-way religion classification. We follow the same pre-processing steps as before. However, we now drop duplicates in name, parent/spouse's name, and religion and split the data into training, validation, and test sets in the ratio 70:15:15 due to a relatively small number of observations belonging to Buddhist, Christian, Jain, and Sikh classes. We follow the same methodology as before for all the models.[7]

Table 11 report the results for the REDS test set. Panel A shows the results when predicting religion using only a single name. The coverage for Name2community is less than 60%. When we assign the majority religion to the ambiguous predictions of Name2community for tie-breaking, the macro-average F_1 score further falls to 38%. On the other hand, LR, CNN, and SVM have higher accuracy along with 100% coverage. Overall, SVM performs the best and has higher precision and recall than Name2community for all the classes. The language model continues to perform poorly for the multi-way religion classification as well. Though CNN performs well, LR and SVM are significantly more accurate at 1% level of significance and have higher macro-average F_1 scores.

Panel B reports results for the concatenated names models. The results improve, especially for CNN and for minority groups which can benefit from richer data as they comprise a smaller number of observations. The F_1 score for the CNN model is now closer, but still much lower than LR and SVM. The overall accuracy is also lower for the CNN model and the difference is statistically significant at 1% level. However, the recall for Muslim names is slightly better for the CNN model resulting in more balanced predictions. Though the coverage for Name2community increases by 9 percentage points, the macro-average F_1 score is reduced. This again shows that there are limited gains from providing multiple names to Name2community. We note that for all our models, most

7. Tables 13 and 14 describe the hyperparameter search space and the selected hyperparameters respectively.

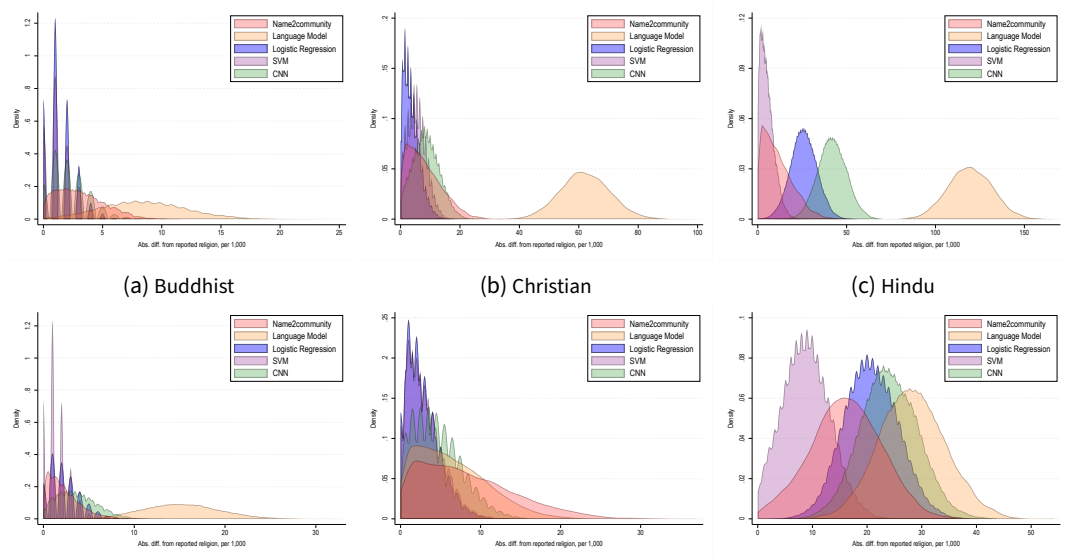
Table 12. Confusion Matrices for multi-religion SVM. Single name (left) and concatenated (right) SVM models on REDS test set. B = "Buddhist", C = "Christian", H = "Hindu", J = "Jain", M = "Muslim", S = "Sikh"

		Predicted						
		B	C	H	J	M	S	Total
True	B	49	0	24	0	0	0	73
	C	0	269	147	0	5	0	421
	H	6	59	14,160	17	55	243	14,540
	J	0	0	30	22	0	0	52
	M	0	1	105	0	1,464	0	1,570
	S	0	2	74	0	15	460	551
	Total	55	331	14,540	39	1,539	703	17,207

		Predicted						
		B	C	H	J	M	S	Total
True	B	52	0	21	0	0	0	73
	C	0	291	125	0	5	0	421
	H	4	33	14,370	7	37	89	14,540
	J	0	0	27	25	0	0	52
	M	0	0	60	0	1,507	3	1,570
	S	0	3	64	0	12	472	551
	Total	56	327	14,667	32	1,561	564	17,207

of the incorrect classifications for Buddhists, Christians, Jains, and Sikhs end up in the Hindu class. This not only reaffirms that Buddhist, Hindu, Jain and Sikh names have common linguistic origins, but also that Christian converts in India often retain their original Hindu names. To illustrate, we show confusion matrices for SVM models in Table 12.

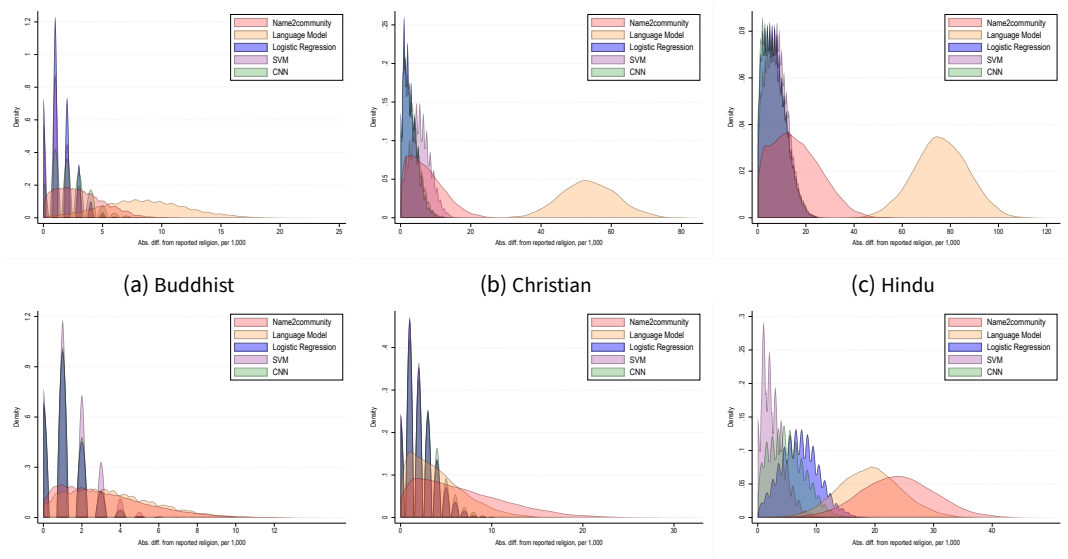
In Figure 10 we show absolute difference between actual and estimated religion counts for single name models—based on 10,000 bootstrap samples with sample size of 1,000 per draw. We find that language model approach performs the worst in estimating aggregate religious composition. On the other hand, SVM performs the best for all the religions. The difference in median accuracy between SVM and Name2community is 2.00, 1.97, 4.62, 0.97, 5.28, and 7.14 per 1,000 for Buddhists, Christians, Hindus, Jains, Muslims, and Sikhs respectively. In Figure 11, we show the results for concatenated names model and find that all our classifiers are better at estimating aggregate religious shares than Name2community and language model. The corresponding median differences in accuracy between SVM and Name2community for concatenated models are 2.44, 1.35, 7.83, 1.67, 3.71, and 21.90 per 1,000. These differences are again sizeable and show the improvement over the baselines in estimating aggregate religious compositions as well. Considering the performance at predicting individual religion and aggregate religious composition, we again find that SVM has the best overall performance. However, as discussed before, when probabilities are of interest LR should be preferred as SVM doesn't directly give probabilities.



(a) Buddhist (b) Christian (c) Hindu

(d) Jain (e) Muslim (f) Sikh

Figure 10. The figure shows the density plot of the absolute difference between reported and estimated religious counts per 1,000 people for the single name models based on the REDS test sets.



(a) Buddhist (b) Christian (c) Hindu

(d) Jain (e) Muslim (f) Sikh

Figure 11. The figure shows the density plot of the absolute difference between reported and estimated religious counts per 1,000 people for the concatenated names models based on the REDS test sets.

Table 13. Hyperparameter Range for Multi-Religion Classifiers. This table presents the range of hyperparameters considered for training our multi-religion classifiers. The final hyperparameters are selected based on evaluation results on REDS validation set.

Parameter	Experimental Range
Embedding dimension	5-80
CNN kernel sizes (k)	1–7
CNN filters	0–300
Dense units	0–400
Dense activation	ReLU, tanh, sigmoid
CNN activation	ELU, ReLU, tanh
Embedding dropout	0–0.25
Dropout	0–0.5
Loss	binary cross-entropy, focal-loss
Batch size	32–1024
Optimizer	Adam, Nadam, rmsprop
Kernel initialization	He uniform, Glorot uniform, He Normal
Minimum learning rate	1×10^{-5} –1×10^{-3}
Batch normalization	True, False
Learning rate reduction factor	0.5–0.8
Patience	3–5 epochs
epochs	20–80
l2 regularization parameter	0–100
TF-IDF Max n-grams	1–12

Table 14. Hyperparameter Choice for Multi-Religion Classifiers. This table lists the hyperparameters used to train our final multi-religion classifiers. All results reported on held-out test sets use these models.

Parameter	Single Name	Concatenated
Embedding dimension	29	30
CNN kernel sizes (k)	{1,2,3,4,5,6,7}	{1,2,3,4,5,6,7}
CNN filters	{50,300,305,200,250,200,200}	{239,248,100,150,150,250,200}
Dense units	400	200
Dense activation	sigmoid	sigmoid
CNN activation	tanh	ELU
Embedding dropout	0.01	0.02
Dropout	0.2	0.2
Loss	binary cross-entropy	binary cross-entropy
Batch size	512	512
Optimizer	Nadam	Nadam
Kernel initialization	He uniform	Glorot uniform
Minimum learning rate	0.0002	0.00027
Batch normalization	True	True
Learning rate reduction factor	0.5	0.5
Patience	2 epochs	3 epochs
epochs	80	60
l2 regularization parameter	64.49 (LR), 79.53 (SVM)	33.39 (LR), 8.47 (SVM)
TF-IDF Max n-grams	12 (LR), 11 (SVM)	10 (LR, SVM)

References

Chaturvedi, S., S. Das, and K. Mahajan. 2021. "The Importance of Being Earnest: What Drives the Gender Quota Effect in Politics?" *Available at SSRN 3962068.*

Clark, J. T., J. A. Curiel, and T. S. Steelman. 2021. "Minmaxing of Bayesian Improved Surname Geocoding and Geography Level Ups in Predicting Race." *Political Analysis,* 1–7.

Dozat, T. 2016. "Incorporating nesterov momentum into adam.(2016)." *Dostupné z: http://cs229. stanford. edu/proj2015/054_report. pdf.*

Elman, J. L. 1990. "Finding structure in time." *Cognitive science* 14 (2): 179–211.

Hochreiter, S., and J. Schmidhuber. 1997. "Long short-term memory." *Neural computation* 9 (8): 1735–1780.

Imai, K., and K. Khanna. 2016. "Improving ecological inference by predicting individual ethnicity from voter registration records." *Political Analysis* 24 (2): 263–272.

Kim, Y., Y. Jernite, D. Sontag, and A. M. Rush. 2016. "Character-aware neural language models." In *Thirtieth AAAI Conference on Artificial Intelligence.*

Srivastava, R. K., K. Greff, and J. Schmidhuber. 2015. "Training very deep networks." In *Advances in neural information processing systems,* 2377–2385.

Susewind, R. 2015. "What's in a name? Probabilistic inference of religious community from South Asian names." *Field Methods* 27 (4): 319–332.