Predicting the gender of Indonesian names

Ali Akbar Septiandri

Titik Koma Digital, Jakarta, Indonesia

E-mail: ali@titikkoma.tech

Abstract. We investigated a way to predict the gender of a name using character-level Long-Short Term Memory (char-LSTM). We compared our method with some conventional machine learning methods, namely Naïve Bayes, logistic regression, and XGBoost with n-grams as the features. We evaluated the models on a dataset consisting of the names of Indonesian people. It is not common to use a family name as the surname in Indonesian culture, except in some ethnicities. Therefore, we inferred the gender from both full names and first names. The results show that we can achieve 92.25% accuracy from full names, while using first names only yields 90.65% accuracy. These results are better than the ones from applying the classical machine learning algorithms to n-grams.

1. Introduction

Inspired by Liu and Ruths [1], we introduced a character-based approach to inferring one's gender from their name. In this study, we focused on Indonesian names. One of the main distinction between Indonesian names and English names is that it is not common to use family names as the surname. Therefore, we compared the two approaches, using only first names and full names, aspiring to get a better view to this problem.

In Indonesian names, we also face challenges like unisex names. For instance, "Dwi", "Tri", or "Rizki" and their variations can be used by either males or females. On the other hand, there are also some unmistakable names, such as "Putra" for males and "Putri" for females. Since Indonesian names come from several different roots, e.g. Sanskrit, Arabic, English, and Chinese, we might encounter a combination of names with those origins. Having said that, some people in Indonesia only have forename without a surname.

In this paper, we employed char-LSTM, a neural network architecture that process a sequence of characters, as a novel approach to tackle this classification problem and compared the result to the ones we got from conventional machine learning algorithms, such as logistic regression, Naïve Bayes, and XGBoost. We applied those algorithms to our self-labeled dataset. We focused only on gender classification in this study aspiring to get the best model to be applicable to cases where it can be useful to infer somebody's gender from his/her name. Examples include better customer interaction and estimating gender-related demographics in social media.

2. Related Work

This work is mainly based on [1]. Although there are many gender classification studies driven by the emergence of social media like Twitter, this prediction task with names as one of the attributes is not common. Some of the work that is related to this task is [2], [3], and [4].

Liu and Ruths [1] built a gender-name association score from first names in their study and stated that this is "a promising approach since the measure compactly encapsulates a prior on the gender of the individual based on their name." In [3], first names are proposed as one of the attributes, but they convert the names to phoneme sequences and then generate n-grams of the phoneme sequences. On the other hand, Burger et al. chose to create 1-5 character n-grams and word unigram from full names [2]. Bergsma et al. also tried binary 1-4 character n-grams and used first names and last names binary token [4].

We believed that the success of LSTM [13] in several complex cases, such as shown in [5] and [6], and neural networks in general could be beneficial in this rather simple study. By doing this, there is no need to extract n-grams from the names. We also explored character-level embedding inspired by [7] yearning for better results. The idea of the char-LSTM architecture is based on the IMDB sentiment classification task provided in [8].

3. Methodology

3.1. Dataset

The names and associated gender data used in this study are compiled from graduate tracer studies. The resulting dataset consists of 4580 males (66.56%) and 2301 females (33.44%), for a total of 6881 names. The dataset is not released as part of this study as there is an obvious privacy concern regarding identifiable names and gender.

We acknowledge that there are certainly biases inherent in the data collection process, e.g. gender proportion and "style" of the names themselves do not reflect Indonesia's general population. However, the methodology of this study can still be applied and further research using more representative dataset is recommended.

3.2. Features

First of all, we removed some symbols, e.g. apostrophe ('), period (.), and hyphen (-), in the names. We also set the names to be written in lowercase characters. This is because it is very rare to find Indonesian names with mixed case.

Aside from the 2-5 character n-gram features as shown in prior work, we proposed simpler features, such as the first and last characters of first and last names that we call *basic* features. We will see later in this study that by using these basic features, we can already achieve fair results. For these features, we transformed them using one-hot-encoder before using them as the classifiers' attributes.

For the 2-5 n-grams, we chose only 1000 top features with the highest values of chi-squared statistic test. This was done to remove n-grams "that are most likely to be independent of class" [9]. Also, this can make the learning process faster.

For the char-LSTM classifier, we extracted the index for every character in the names. This will yield lists of characters with different length. Therefore, we padded the lists with zeros until they have the same length as the maximum number of characters in the dataset, which in our case is 56. For first names only, the maximum number of characters is 17. The distribution of the names by the number of characters can be seen in Figure 1.

3.3. Algorithms

We employed neural networks implementation known as Keras [8] in our experiments. Using embedding layer and LSTM, we passed the inputs and then use simple logistic regression as the output layer. We trained this model with Adam [12] as the optimisation routine using default parameters and mini-batch size 32. We also tried different number of hidden units for the embedding layer and LSTM. For each layer, we tried 64, 128, and 256 for full names, and 32, 64, and 128 for first names.

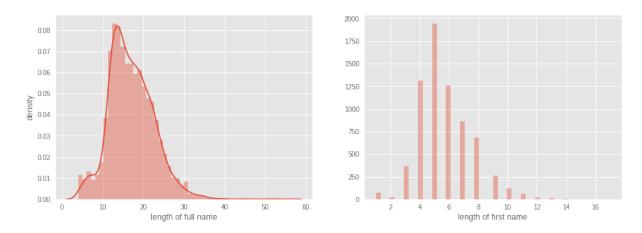


Figure 1. Length of the names

On the other hand, we used scikit-learn [9] for the conventional machine learning algorithms other than XGBoost. We chose Naïve Bayes for its simplicity (almost no hyperparameter tuning needed) [11] and logistic regression as the "simpler version" of neural networks. XGBoost [10], however, was chosen because of its decent performance in lots of machine learning competitions. For the conventional machine learning algorithms, we tuned the hyperparameters using options as shown in Table 1. This was done by grid search and cross-validation.

Classifier	Hyperparameter	Values tested
LogReg	penalty	['11', '12'] [0.01, 0.1, 1, 10, 100]
XGRoost	max_depth	[3, 4, 5, 6, 7, 8, 9, 10]
MGDOOSU	min_child_weight	[0, 0.1, 1, 100.1000]

gamma

[0, 0.1, 1, 100, 1000]

Table 1. List of hyperparameters

3.4. Evaluation Approach

Since we found this dataset to be unbalanced, aside from accuracy, we calculated the F_1 -score as well. F_1 -score is defined as:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

We tried the algorithms on features from full names. Our goal is to produce the best model possible from these features. Then, we attempted to use first names only as the features to see whether we would be able to achieve similar results. In the first experiment with LSTM, we set the number of the epoch to be 20.

4. Results

4.1. Inferring Gender from Full Names

Using the tuned algorithms with basic features and n-grams, we got the results as shown in Table 2. We can see that we achieved the best performance for Naïve Bayes and logistic regression

using 3-gram features, while using 2-gram shows a slightly better result for XGBoost.

	Accuracy (%)			F_1 (%)		
Features	NB	LogReg	XGBoost	NB	LogReg	XGBoost
basic	78.69	80.44	81.50	84.07	85.59	86.35
2-gram	79.81	84.21	83.87	85.10	88.63	88.55
3-gram	84.12	$\bf 85.28$	83.39	88.35	89.44	88.30
4-gram	83.78	83.54	79.90	88.52	88.33	86.57
5-gram	80.77	80.92	76.71	86.93	86.96	85.22

Table 2. Results from basic and n-gram features

In the next step, we trained the char-LSTM model using different number of hidden units. By doing this, we got two best results with slightly different perforamance. The best one we got is from using \mathbb{R}^{256} vector embedding and 64 hidden units for LSTM, while the second best is from vector embedding in \mathbb{R}^{128} and 256 hidden units for LSTM. The results as well as the effect of the number of epochs to the accuracy can be seen in Figure 2. The accuracy and F_1 for the best model are 92.25% and 94.35% respectively, while the second best model yields 92.20% accuracy and 94.31% F_1 .

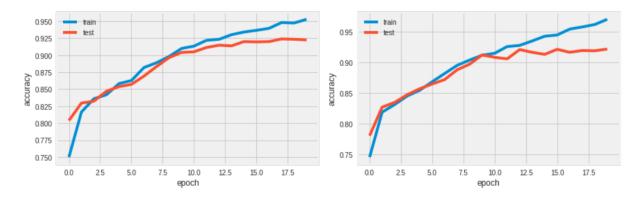


Figure 2. The effect of epoch to accuracy for the best (left) and second best (right) models

4.2. Inferring Gender from First Names Only

Similar trends can be seen in the results from features extracted from first names. As we can see in Table 3, the 3-gram feature outperforms other features when using Naïve Bayes and logistic regression. On the XGBoost case, 2-gram yields better result than 3-gram which is similar to the result from using full names.

The results with char-LSTM on first names are also promising. The best model in this experiment was from using \mathbb{R}^{128} vector embedding and 128 hidden units for LSTM. With this architecture, we reached **90.65**% accuracy and **93.19**% F_1 . This means that the F_1 is about 1.16% off from the best model produced with full names. The effect of epoch in the best model from first names can be seen in Figure 3.

4.3. Exploration on Specific Names

Since we have seen that the best model came from the char-LSTM algorithm, we wanted to know how this model performs in several cases. One of the advantages of using the char-LSTM

Table 3. Results from basic and n-gram features of first names

	Accuracy (%)			F_1 (%)		
Features	NB	LogReg	XGBoost	NB	LogReg	XGBoost
basic	74.77	75.30	76.85	81.01	81.73	83.31
2-gram	78.64	81.60	$\bf 82.37$	84.51	87.27	87.80
3-gram	79.85	81.50	78.35	85.95	87.17	85.85
4-gram	77.34	77.38	72.74	85.06	85.18	83.06
5-gram	72.59	72.69	69.93	82.93	83.02	81.80

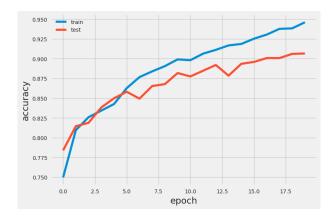


Figure 3. The effect of epoch to accuracy for the best model using char-LSTM

is that you can see the changes in the class probability for every character addition. Therefore, we fed the best model trained with full names to predict some of the names. Bear in mind that these names can be in the first, middle, or last part of the names.

The model we used in this experiment is the one with 256 and 64 hidden units for embedding and LSTM layers respectively. Probabilities produced from this model show the likelihood of whether the name is a male. Therefore,

$$P(y = \text{female}|\text{name}) = 1 - P(y = \text{male}|\text{name})$$

We evaluated the name 'Putra' and 'Putri'. As mentioned before, these names are almost unambiguous, i.e. 'Putra' is for males and 'Putri' is for females. In the dataset, there are 215 people with 'putra' in their name and 100 females have 'putri' in their name. Only one name that contains 'Saputra' and is a female. The evaluation per one character increment can be seen in Figure 4. The blue and pink bars represent the probability of the name being male or female respectively.

5. Conclusions and Future Work

Our work has shown that using char-LSTM can improve the model for predicting gender from names compared to only using conventional machine learning algorithms with n-grams. We managed to increase the best accuracy from 3-gram using logistic regression, i.e. 85.28%, to 92.25% using char-LSTM. This means that we reduced the remaining error rate by 47.35%. The difference between the best F_1 -score from n-grams and char-LSTM in this study is 4.91% (from 89.44% to 94.35%).

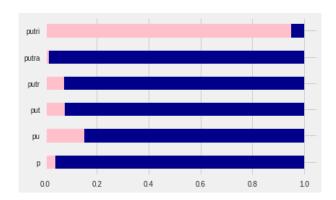


Figure 4. Char-LSTM evaluation on specific names

In the second experiment, we found out that using first names only might be enough. With even fewer features, the char-LSTM could produce 90.65% accuracy. This level of accuracy also outperformed the results from n-grams. However, the first experiment suggested that using full names is better if possible, i.e. when the surnames could be gender-specific.

There are several ways to improve this study, e.g. using different nationalities of the names or using usernames from social media. We would also like to see how this approach could improve gender prediction in general given other features from social media. Furthermore, we believe that these methods can be extended to predicting other things, such as the ethnicity of the people. It would also be interesting to build a generative model, e.g. using conditional Generative Adversarial Networks, to generate names given the gender or the ethnicity.

Acknowledgement

The author would like to thank Okiriza Wibisono and Yosef Ardhito Winatmoko for their advice and support to this study.

References

- [1] Liu W and Ruths D 2013 AAAI spring symposium: Analyzing microtext vol 13 p 01
- [2] Burger J D, Henderson J, Kim G and Zarrella G 2011 Proceedings of the Conference on Empirical Methods in Natural Language Processing (Association for Computational Linguistics) pp 1301–1309
- [3] Alowibdi J S, Buy U A and Yu P 2013 2013 12th International Conference on Machine Learning and Applications (IEEE) pp 365-369 ISBN 978-0-7695-5144-9 URL http://ieeexplore.ieee.org/document/ 6784644/
- [4] Bergsma S, Dredze M, Van Durme B, Wilson T and Yarowsky D 2013 HLT-NAACL pp 1010-1019
- [5] Sutskever I, Martens J and Hinton G E 2011 Proceedings of the 28th International Conference on Machine Learning (ICML-11) pp 1017–1024
- [6] Graves A 2013 arXiv preprint arXiv:1308.0850
- [7] Kim Y, Jernite Y, Sontag D and Rush A M 2015 arXiv preprint arXiv:1508.06615
- [8] Chollet F et al. 2015 Keras https://github.com/fchollet/keras
- [9] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M and Duchesnay E 2011 Journal of Machine Learning Research 12 2825–2830
- [10] Chen T and Guestrin C 2016 Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD '16 (New York, NY, USA: ACM) pp 785-794 ISBN 978-1-4503-4232-2 URL http://doi.acm.org/10.1145/2939672.2939785
- [11] Rish I 2001 IJCAI 2001 workshop on empirical methods in artificial intelligence vol 3 (IBM New York) pp 41–46
- [12] Kingma D and Ba J 2014 arXiv preprint arXiv:1412.6980
- [13] Hochreiter S and Schmidhuber J 1997 Neural Computation 9 1735–1780