Indonesian name matching using machine learning supervised approach

1. Mohamad Alifikri
   School of Computing, Telkom University, Indonesia
   alifikri@gmail.com

2. Moch. Arif Bijaksana,
   School of Computing, Telkom University, Indonesia
   arifbijaksana@telkomuniversity.ac.id

Abstract. Most existing name matching methods are developed for English language and so they cover the characteristics of this language. Up to this moment, there is no specific one has been designed and implemented for Indonesian names. The purpose of this thesis is to develop Indonesian name matching dataset as a contribution to academic research and to propose suitable feature set by utilizing combination of context of name strings and its permute-winkler score. Machine learning classification algorithms is taken as the method for performing name matching. Based on the experiments, by using tuned Random Forest algorithm and proposed features, there is an improvement of matching performance by approximately 1.7% and it is able to reduce until 70% misclassification result of the state of the arts methods. This improving performance makes the matching system more effective and reduces the risk of misclassified matches.

1. Introduction
Name matching methods can determine the degree of relatedness between person names and provide score for the strength of the match. The higher the match score, the more likely it is that the records relate to the same individual. However, many existing name matching methods tend to be generic and language or cultural independent. As a result, one method could be best applied in certain country/culture but in other country, it does not always show the best performance.

The mainstream approaches for name matching problem are using phonetic encoding and pattern matching. Further techniques have been developed for both approaches beyond the original ones, even combination of the two with the aim to improve the matching quality. The techniques use similarity measure rather than identity, such as edit distance, n-gram matching and soundex.

Despite extensive exploration of this task, most works have focused on languages, particularly in English. Phonetic matching method like Soundex, Metaphone, Double Metaphones developed using phonetic for English names primarily [1]. Pattern Matching methods such as Edit Distance is language independent, but too general; it does not capture the majority of possible variations in specific culture or country. For example: existing methods could not capture Indonesia names: Cecep and Tjetjep, as a
matching name pair. This presents a new challenge for name matching particularly in specific country like Indonesia.

One of the main obstacles to perform name matching task for Indonesia names is that there is no available dataset of Indonesia names which are naturally formed into pairs exist and ready for the research. This research beside propose a solution which include selection of machine learning classification algorithm and suitable features that have best performance on matching Indonesia names, also develop representative Indonesia name pair dataset as a basis of current and future name matching research.

2. Review of Literature

2.1. Name Matching Variations

Most works with respect to name matching focus on cross language domain. This is justifiable due to the most difficult issue when dealing with distinctive languages which have diverse characteristics, and require alignment using transliteration, even custom one, to do name matching task. In any case, the origin of name matching issue is from the same fundamental issue, that is name variations. One work about name variations was reported by [2] in her report describing all variations commonly appeared in Indonesia names. Another reference [3] divide name variations into 2 group of categories which are linguistic (transcription variants, homophones, name alias) and non-linguistic (the use of initials, change order of name parts, spelling error).

2.2. Name Matching Methods

Many variations for approximate string matching have been developed, there are two main categories [1][4]; pattern matching and phonetic matching techniques. Some of most commonly used techniques are pattern and phonetic matching.

Name matching is also considered as a classification task [5][6]. Each string pair is an instance: a positive classification means that two strings can refer to the same name. When the compared candidate name pairs are only classified into matches and non-matches (but not potential matches), then this classification is known as a binary classification problem. Further, if preparing training set in the form of name pairs with their label match or non-match, at that point a supervised classification approach can be utilized to train a classification model. The prepared model is then utilized to distinguish record pairs with an unknown status into matches or non-matches. Three popular supervised classification techniques that have been employed in the area of data matching and deduplication are Linear SVM, Logistic Regression and Random Forest. They are used to evaluate name matching performance during this thesis research.

3. Research Methodology
There are four stages use in this research as depicted in Figure 1:

![Figure 1. Experiment Scenario](image)

### 3.1. Data Preparation & Dataset Creation

This research started with collecting Indonesian names from two online sources by performing web crawling to obtain the person names data which are list of high school student names (taken from Data Pokok Pendidikan SMA-SMK website) and constituent data of Pilkada 2017 (Simultaneous Regional Election) organized by KPU. Total number of Indonesia person names collected are 2,125,210 names. This stage was conducted by running series of data cleaning and standardization process which are stripping whitespace, removing honorific/titles, duplication/missing values handling.

Concerning pairwise format, in binary classification problem, the dataset was commonly categorized into two types. They were positive and negative examples (instances). Positive examples are pair of names that considered as matching/similar names, on the contrary negative are dissimilar pair. Data set generation stage was performed with goal to compose dataset containing both positive and negative examples as shown in Table 1.

### Table 1. Data preparation result

| Jarowinkler_score | Name1               | Name2               | Pairname | Variant_type       | label |
|-------------------|---------------------|---------------------|----------|--------------------|-------|
| 0                 |                      |                     |          |                    |       |
| 0.576923          | Dhiyan Prianto      | Sofiyani            | Dhiyan Prianto, Sofiyani |      | 0     |
| 0.867530          | Syathri Syahrozi    | Syahrozi            | Syahrozi, Syahrozi, Syahrozi | Word addition/removal | 1     |
| 0.595238          | Mustofa             | Lusi                | Mustofa, Lusi |                | 0     |
| 0.891429          | Siti Arii           | Siti A.             | Siti Arii, Siti A | Short form (Initial) | 1     |
| 0.750333          | Rudy Supriyanta     | Supriyanta          | Rudy Supriyanta, Supriyanta |       | 0     |
| 0.526385          | Grace Nathascha     | Thascha             | Grace Nathascha, Thascha |       | 0     |
| 0.940308          | Intan Purnama Sari  | Eranus              | Intan Purnama Sari Eranus, Intan Purnama Sari Eranus | Type random space deletion | 1     |
| 0.869474          | Rahan Fathurrahman  | Lasn            | Rahan Fathurrahman, Rahan Fathurrahman | Transcription variants | 1     |
| 0.849591          | Yohan Hendrik Manior| Yohan H.             | Yohan Hendrik Manior, Yohan H. | Short form (Initial) | 1     |
| 0.890009          | Yuyun Jufri         | Yuyun              | Yuyun Jufri, Yuyun |                     | 1     |

### 3.2. Experiment 1 Baseline

This experiment utilized pattern and phonetic matching methods as the baseline of experiment. The goal is to setup the baseline performance compared to our proposed method. There are 30 different pattern matching method, from winkler, permute-winkler, bag distance, edit distance, until qgram, etc.
And for phonetic matching methods, there are 15 different methods (soundex, phonex, phonix, caverphone, until koelner).

Name matching was performed by comparing the score of each method with their best threshold. The score below the threshold was non-matching pair, and otherwise was matching pair. This classification was then compared with actual label of each pair and count how many pair were correctly or incorrectly classified. Using F1 measure metric, this experiment then determined how strong the predictive power for each tested method.

The best performing overall matching algorithm for Indonesia pair names dataset was achieved by permwinkler with F1 score 0.972. However, if name matching requires both accuracy and speed, then the method best suited this need is qgram2P. In general, pattern matching algorithm beat phonetic matching algorithm in term of performance (F1 measure score).

3.3. Experiment 2 (Feature Extraction)

This experiment focused on finding the way to improve the performance of the baseline methods using Machine Learning classification methods, considering that the opportunity area could be improved from those identified on previous experiment. This experiment selected three classifiers to conduct classification, which were Linear SVM (SVM), Logistic Regression (LR) and Random Forest (RF).

The experiment 1 test through some of pattern matching methods, phonetic matching methods, and each of them evaluated one-by-one as single feature by machine learning algorithm. List of features can be observed in Table 2. Beside using existing pattern and phonetic methods, the research also introduced and proposed new feature group named context_feature.

![Figure 2. Feature extraction experiment](image)

As shown in **Figure 2**, after experiment 1 ran, the best single feature which had the best F1 score was selected. It was then passing through to experiment 2. This experiment 2 testing used combination of 2 features, consisted of one feature obtained from experiment 1, and another one was a new additional feature. The combination of 2 features that gave the best F1 score was then used as baseline for experiment 3. In experiment 3, an additional feature was added to result of experiment 2 to form 3 features. The combination of 3 features that gave the maximum F1 score were then selected as the final and the best feature set of this experiment.

The goal of this experiment was to find what feature group contribute the most significant to the F1-score performance. It was indicated from the result (**Table 3**) that the best feature group is achieved by using context_features+permwinkler+ontolcs3avg with F1-score 0.993. However, this score is the same with previous experiment that using only 2 features context_features+permwinkler or context_features+lcs3avg. By this result it can be concluded that the increasing number of feature
is not always making the performance better anymore. Another important result is among 3 classifiers, the best classifier was achieved by \textbf{Random Forest}.

\textbf{Table 2. List of features for feature extraction experiment}

| No. | Feature group | Feature type | Permitted value | Description |
|-----|---------------|--------------|-----------------|-------------|
| 1   | permwinkler   | numerical    | 0.0 – 1.0       | Winkler permutation score |
| 2   | lcs2          | numerical    | 0.0 – 1.0       | Longest common substring with minimum length of substrings 2 |
| 3   | lcs3          | numerical    | 0.0 – 1.0       | Longest common substring with minimum length of substrings 3 |
| 4   | ontolcs3      | numerical    | 0.0 – 1.0       | Ontology longest common substring with minimum length of substrings 3 |
| 5   | editex        | numerical    | 0.0 – 1.0       | Phonetic aware edit-distance |
| 6   | qgram2        | numerical    | 0.0 – 1.0       | q-grams of length 2 |
| 7   | qgram3        | numerical    | 0.0 – 1.0       | q-grams of length 3 |
| 8   | qgram2P       | numerical    | 0.0 – 1.0       | Padded q-grams of length 2 |
| 9   | qgram3P       | numerical    | 0.0 – 1.0       | Padded q-grams of length 3 |
| 10  | tfidf         | numerical    | 0.0 – 1.0       | Cosine similarity of TF-IDF character 1,2,3,4-gram |
| 11  | nysiis4       | numerical    | 0.0 – 1.0       | NYSIIS limited/padded to length 4, use 5 sets of rules to convert name into code |
| 12  | phonex4       | numerical    | 0.0 – 1.0       | Phonex limited/padded to length 4, combination of Soundex and Metaphone |
| 13  | phonix4       | numerical    | 0.0 – 1.0       | Phonix limited/padded to length 4, applies string standardisation rules before a nine numerical code transformation |
| 14  | soundex4      | numerical    | 0.0 – 1.0       | Soundex limited/padded to length 4 |
| 15  | context features | categorical | string and T/F (True or False) | The number of features in this group is variable, depend on how many word on string name. The features are extracted per word. |
| 1   | extract prefix/suffix features | | | |
| 2 | insert_char_position (start/mid/end) |
| 3 | insert_is_char_a_consonant (T/F) |
| 4 | insert_is_with_rightchar_ejaanlama (T/F) |
| 5 | delete_char_position (start/mid/end) |
| 6 | delete_is_char_a_consonant (T/F) |
| 7 | delete_is_with_rightchar_ejaanlama (T/F) |
| 8 | replace_char_position (start/mid/end) |
3. Extracting context features
Six features are extracted following the idea from what kind of misclassification occurred:
- context_same_last_char (T/F)
- context_singleword (T/F)
- context_string1_in_string2 (T/F)
- context_difference_stringlength (T/F)
- context_differ_only_last_char (T/F)
- context_same_truncated_words_remove_space_between_words (T/F)

### Table 3. Result of feature extraction experiment using 3 features

| Feature Group | FN (SVM) | FP (SVM) | Misclassification # features | F1 on testing set |
|---------------|---------|---------|----------------------------|-------------------|
| context_features| 9       | 8       | 6                          | 0.987             |
| context_features| 12      | 7       | 11                         | 0.988             |
| context_features| 13      | 10      | 9                          | 0.988             |
| context_features| 9       | 11      | 10                         | 0.988             |
| context_features| 9       | 9       | 12                         | 0.988             |
| context_features| 10      | 10      | 11                         | 0.988             |
| context_features| 10      | 11      | 12                         | 0.988             |
| context_features| 13      | 11      | 12                         | 0.988             |
| context_features| 13      | 11      | 12                         | 0.988             |
| context_features| 13      | 11      | 12                         | 0.988             |
| context_features| 11      | 10      | 10                         | 0.988             |
| context_features| 11      | 10      | 10                         | 0.988             |

3.4. Experiment 3 Benchmark Classifier

The purpose of this experiment was to provide brief snapshot on the performance of multiple classifiers using their default parameters to get the idea on what was the best performing classifier to solve problem of name matching.

### Table 4. Experiment 3 result applying multiple classifiers and using context + permwinkler features

| Model                | F1 Score  | Process Time |
|----------------------|-----------|--------------|
| Logistic Regression  | 0.909009  | 0.439647     |
| AdaBoost             | 0.988262  | 5.564991     |
| NeuralNet            | 0.888628  | 17.073669    |
| Random Forest        | 0.988261  | 0.663623     |
| Perceptron           | 0.987778  | 0.865396     |
| Linear SVM           | 0.987624  | 8.571252     |
| Decision Tree        | 0.985388  | 0.876483     |
| Stochastic Gradient Descent | 0.984654 | 0.181748     |
| Bernoulli Naive Bayes| 0.977275  | 0.273834     |
| KNN                  | 0.972321  | 6.108816     |
4. Result and Discussion

From all experiments conducted on each method either using baseline or machine learning classification, the result could be compared by looking on Table 5.

Table 5. Comparison of experiment results

| Experiment                          | Methods          | TP  | TN  | FP  | FN  | F1      | #Samples | Error | Ratio error |
|-------------------------------------|------------------|-----|-----|-----|-----|---------|----------|-------|-------------|
| Baseline                            | permwinkler      | 853 | 810 | 23  | 20  | 0.976   | 1,716    | 43    | 2.51%       |
| Classification machine learning     | context features | 862 | 816 | 7   | 20  | 0.985   | 1,716    | 27    | 1.57%       |
| Classification machine learning     | context features+permwinkler | 875 | 828 | 5   | 5   | 0.993   | 1,716    | 13    | 0.76%       |

- Based on comparison of the result, it is revealed that machine learning classification have shown better performance for name matching task compared to what can be best achieved by baseline or conventional method (using pattern matching techniques).
- Machine learning classification algorithms have wide range of classifier algorithms and parameters that could be tuned in order to find what is the best one. Experimental evaluation in this thesis shows that by using proposed features (context features) and selecting the best classifier (Random Forest), shows that the performance could be improved by 0.9% (using context features only) and 1.7% (by using context and permwinkler features), compared to baseline. This small improvement on F1 measure is due to baseline already have very good number of F1 (F1 = 0.976).
- Context features and combination of context features and permwinkler can reduce many misclassifications significantly by 70% reduction that present on baseline permwinkler.

5. Conclusion and Future Works

5.1. Conclusion

Indonesia name dataset by this thesis is developed. This dataset hopefully will open more research for improving Indonesian name matching system. From the experiments performed for name matching of Indonesian names using this dataset, it is concluded that machine learning supervised classification techniques have good chance to outperformed the state of the art using conventional pattern matching techniques. The key importance aspect for achieving superior performance is on how feature constructed and hyperparameter tuning of machine learning classification algorithms.

Although conventional method had achieved near perfect performance (high F1 score), this research had shown that by using context features combined with other selected feature (permwinkler), it could surpass the best achieved performance of conventional pattern matching techniques (permute-winkler).

5.2. Recommendation

It is advisable to provide more comprehensive solution, that future research can be covered the other source of name variation which is due to name alias.

Thorough analysis on Indonesia names variation could be improved by focusing on majority of Indonesia population which is Java tribe or others. By performing deep analysis, hopefully one can identify better predictive and discriminative features to improve the performance of name matching task.

It has also the possibility for improvement if the misclassification result can be analyzed and combined further using heuristic rule to make additional classification.
References

[1] P. Christen, “A Comparison of Personal Name Matching: Techniques and Practical Issues,” *Data Min. Work. 2006. ICDM Work. 2006. Sixth IEEE Int. Conf.*, no. September, pp. 290–294, 2006.

[2] D. Isyanti, “Changes in the Rules for Indonesian Name Personal Headings,” 2006.

[3] B. Lisbach and V. Meyer, *Linguistic Identity Matching*. Springer Science & Business Media, 2013.

[4] C. Varol and C. Bayrak, “Hybrid Matching Algorithm for Personal Names,” *ACM J. Data Inf. Qual.*, vol. 3, no. 4, pp. 1–18, 2012.

[5] W. W. Cohen, P. D. Ravikumar, S. E. Fienberg, P. D. Ravikumar, and S. E. Fienberg, “A Comparison of String Distance Metrics for Name-Matching Tasks,” *Proc. IJCAI-03 Work. Inf. Integr. Web*, pp. 73–78, 2003.

[6] P. Christen, *Data Matching Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer Berlin Heidelberg, 2012.