An Improved Indonesian Grapheme-to-Phoneme Conversion Using Statistic and Linguistic Information

Agus Hartoyo, Suyanto

Faculty of Informatics - IT Telkom, Jl. Telekomunikasi No. 1 Terusan Buah Batu
Bandung, West Java, Indonesia
truegushar@yahoo.co.id, suy@ittelkom.ac.id

Abstract. This paper focuses on IG-tree + best-guess strategy as a model to develop Indonesian grapheme-to-phoneme conversion (IndoG2P). The model is basically a decision-tree structure built based on a training set. It is constructed using a concept of information gain (IG) in weighing the relative importance of attributes, and equipped with the best-guess strategy in classifying the new instances. It is also leveraged with two new features added to its pre-existing structure for improvement. The first feature is a pruning mechanism to minimize the IG-tree dimension and to improve its generalization ability. The second one is a homograph handler using a text-categorization method to handle its special case of a few sets of words which are exactly the same in spelling representations but different each other in phonetic representations. Computer simulation showed that the complete model performs well. The two additional features gave expected benefits.

Keywords: Indonesian grapheme-to-phoneme conversion, IG-tree, best-guess strategy, pruning mechanism, homograph handler.

1 Introduction

Many methods of data driven approach was proposed to solve grapheme-to-phoneme (G2P) conversion problem, such as instance-based learning, artificial neural networks, and decision-tree. In [7], it was stated that an IG-tree + best-guess strategy has high performance. It compresses a given training set into an interpretable model. In this research, the method is adopted to develop a new model for Indonesian G2P (IndoG2P). In the new model, two new features for improvement are added: a pruning mechanism using statistic information and a homograph handler based on some linguistic information provided by a linguist.

According to the fact that the model is a lossless compression structure, which means that it stores all data including those of outliers into rules, a pruning mechanism is proposed to prune some rules accommodating outliers. Hence, the model is expected to increase its generalization, but decrease its size.

Furthermore, the model does not handle homograph problems. The letter-based inspection mechanism performed letter by letter internally in a word cannot handle a few sets of words which are exactly the same in spelling representations but different

Received 23/11/09
Accepted 16/01/10
Final version 12/03/10

in phonetic representations. In order to solve this problem, the system should perform an inspection mechanism for wider context. It is clear that the problem is actually only how to recognize the topic of the surrounding text (sentence, paragraph, or passage) which is known as the problem of text categorization. Since [8] stated that the centroid-based classifier for text categorization significantly outperforms other classifiers on a wide range of data sets, the classifier is adopted to solve homograph problem in this research.

2 IndoG2P

The IndoG2P system is designed to use both statistic and linguistic information. This design is expected to solve some different problems in G2P.

2.1 The Phonetic Alphabets

This research uses IPA (International Phonetic Association) Indonesian alphabet system to symbolize pronunciations at the phonetic side of its dataset. As explained in [1], 6 vowels and 22 consonants compose the alphabet system as listed in table 1.

Table 1. The IPA Indonesian alphabets

Phoneme	Category	Sample Word	Phoneme	Category	Sample Word
a	vowel	/akan/	l	consonant	/lama/
e	vowel	/sore/	m	consonant	/makan/
ə	vowel	/ənam/	n	consonant	/nakal/
i	vowel	/ini/	p	consonant	/pintu/
o	vowel	/toko/	r	consonant	/raja/
u	vowel	/baru/	s	consonant	/sama/
b	consonant	/tembakan/	t	consonant	/timpa/
c	consonant	/cari/	w	consonant	/waktu/
d	consonant	/duta/	x	consonant	/axir/
f	consonant	/faksin/	y	consonant	/yakin/
g	consonant	/gula/	z	consonant	/zat/
h	consonant	/hari/	š	consonant	/mašarakat/
j	consonant	/juga/	ŋ	consonant	/təmpuru-ŋ/
k	consonant	/kaki/	ň	consonant	/ňaňian/

2.2 Datasets

The system involves two datasets: 1) IndoG2P datasets used to train the system in building the IG-tree model validate the rules during the IG-tree pruning process, and test the IG-tree classifier as IndoG2P conversion system; and 2) Homograph dataset used to train the centroid-based classifiers and test them as homograph handler.

IndoG2P Datasets. "Given a written or spelled word in Indonesian, the system should output how the word is pronounced" is the main problem IndoG2P conversion must cope with. So, this dataset should be able to give examples for the learning system about how words in Indonesian are spelled and then pronounced. It is simple to understand that the dataset can be a table with two attributes with each record demonstrating spelling and pronunciation of a word: spelling transcription of a word on the first attribute and phonetic transcription of the same word on the second one. The format of the dataset is shown in Table 2.

Table 2. Format of the IndoG2P dataset.

Graphemic transcription	Phonetic transcription
malang	mala-ŋ
tembakan	tembakan
tempurung	təmpuru-ŋ
tempatmu	təmpatmu

The learning mechanism requires that type of the relation between spelling symbols and their corresponding phonetic symbols is one-to-one mapping. It seems to be no problem in case the spelling and phonetic transcriptions of a word is basically in the same length as shown in word "tembakan" (meaning "a shot") and "tempatmu" (meaning "your place"). In case the spelling and phonetic transcriptions of a word basically differ in length, [7] suggests to perform an alignment mechanism as shown in word "malang" (meaning "poor" or "unfortunate") and "tempurung" (meaning "shell" or "skull") by inserting phoneme null '-' at a certain position in the phonetic transcription in such way that (i) every single spelling symbol is mapped to a single phonetic symbol; (ii) its grapheme-to-phoneme mapping is viable (i.e. that they can be motivated intuitively or linguistically); (iii) the combination of all mappings within the alignment has a maximal probability; and (iv) it is consistent with alignments of other similar words.

Table 3. The IndoG2P datasets.

Dataset	Number of instances	Percentage
Training set	5,455	80%
Validation set	679	10%
Test set	679	10%

In this research, IndoG2P dataset is developed from a corpus of words collected from articles published by an Indonesian newspaper, and its grapheme-phoneme pairs' correctness was validated by a professional Indonesian linguist. The dataset consist of 6,791 distinct instances which are then randomly divided into three subsets those are a training set to train the system in building the IG-tree model, a validation set to validate the rules during the IG-tree pruning process, and a test set to test the IG-tree classifier. Proportions of the three subsets are illustrated by Table 3.

Homograph Datasets. These are datasets used in the module of homograph handler. The datasets are composed by texts as their instances. In this module, a particular dataset is provided for a particular homograph word. Any text in a dataset for a homograph word must: 1) contains at least one occurrence of the related homograph word; 2) be composed with relevant sentences; and 3) be labeled with the category representing phonetic representation of the ambiguous graphemes. Our real-world datasets are in this research composed with used texts taken from many articles in the Internet. Referencing to the list of Indonesian homograph words shown in Table 5, we provide 5 datasets for 5 homograph words as follows.

Table 4. Homograph datasets

Homograph word	Training set	Test set
apel	80	20
penanya	14	5
sedan	48	12
mental	40	10
tahu	48	12

2.3 IG-Tree + Best-Guess Strategy

As a lossless compression structure, the origin (without pruning mechanism) IG-tree stores in a compressed format the complete grapheme-to-phoneme knowledge of all words provided by the training set. In this system, compression doesn't only mean the decrease of the model's size, but it means generalization as well. As explained in [7], the generated rules can be seen as optimized, generalized lexical lookup. Words spelled similarly are pronounced similarly since the system's reasoning is based on analogy in the overall correspondence of the grapheme-to-phoneme patterns. The system automatically learns parts of words on which similarity matching can be safely performed. At the end of the learning phase a generated rule actually corresponds to a grapheme with a minimal context which disambiguate mapping of the grapheme into a certain phoneme.

For an illustration, see how the system determines the phonetic representation for grapheme <e> in <tempayan> (meaning "large water jar") when dataset shown in Table 2 is given as its learning materials. Based on the learning materials the system finds that grapheme <e> has two probable phonemic representations, those are /e/ and /ə/. Both maximal subword chunk <tembakan> and <tempatmu> actually disambiguate the <e> mapping patterns, in the meaning that the context surroundings <e> in chunk <tembakan> certainly leads its <e> to be mapped to /e/ in the same way as that in chunk <tempatmu> certainly leads its <e> to be mapped to /ə/.

However the sub-word chunk does not represent minimal context disambiguating the mapping patterns. In contrast, subword chunk <em> represents a smaller context but it is ambiguous since this chunk belongs to some words with different <e>'s phonetic representation. Hence, this system during the learning phase will look for more contextual information and finally find that subword chunk <temb> represents the minimal context disambiguating the focus grapheme <e> to be pronounced as /e/

and <temp> represents the minimal context disambiguating the focus grapheme <e> to be pronounced as /ə/. So, when the system is requested to determine what a certain grapheme in a given word should be pronounced, it will find a mapping pattern on the focus grapheme and matching context, and then get the phonetic label led by the pattern as the answer. In our case since the given word <tempayan> on focus grapheme <e> matches with context represented by subword chunk <temp>, it suggests the system to map the focus grapheme to phoneme /ə/ instead of /e/. When other words such "ditempati" (meaning "being inhabited") and "tempatku" (meaning "my place") are given, the generalization ability of the system is shown, as the same rule covers these cases as well.

Dataset Transformation. On the lowest level, instead of running word by word, IndoG2P conversion actually runs letter by letter. If our problem is considered as a classification problem, given an unknown instance with attributes of a focus grapheme and its context graphemes, IndoG2P is a classification task responsible to label the instance with a phonetic representation. This awareness suggests us to transform the IndoG2P dataset discussed before — a word-by-word dataset, we can say — to its new format of letter-by-letter dataset. The basic idea of the transformation's algorithm is consecutively locating each grapheme (occurred in a words) as the focus / target grapheme and ensuring that when a grapheme is located as focus, other graphemes occurred in the same word are simultaneously located on their appropriate context position. As the number of records belonging to word-by-word dataset is the number of words itself, the number of records belonging to letter-by-letter dataset is the total number of letters occurred in whole words. This transformation is illustrated in Fig. 1.

Graphemictranscription	Phonemic transcription
kamper	kampər
tembak	tembak

L7	L6	L5	L4	L3	L2	L1	F	R1	R2	R3	R4	R5	R6	R7	Phonemic label
.	.	.	.	.	.	^	k	a	m	p	e	r	^	.	k
.	.	.	.	.	^	k	a	m	p	e	r	^	.	.	a
.	.	.	.	^	k	a	m	p	e	r	^	.	.	.	m
.	.	.	^	k	a	m	p	e	r	^	.	.	.	.	p
.	.	^	k	a	m	p	e	r	^	.	.	.	.	.	ə
.	^	k	a	m	p	e	r	^	.	.	.	.	.	.	r
.	.	.	.	.	.	^	t	e	m	b	a	k	^	.	t
.	.	.	.	.	^	t	e	m	b	a	k	^	.	.	e
.	.	.	.	^	t	e	m	b	a	k	^	.	.	.	m
.	.	.	^	t	e	m	b	a	k	^	.	.	.	.	b
.	.	^	t	e	m	b	a	k	^	.	.	.	.	.	a
.	^	t	e	m	b	a	k	^	.	.	.	.	.	.	k>

Fig. 1. Dataset transformation.

As shown in Fig. 1, we provided 14 context graphemes surrounding the focus grapheme; those are 7 for each of right and left side (as R1 represents the first context on the right, L1 represents the first context on the left, and so on). The width of context provided in the dataset should be able to accommodate the context expansion (More about context expansion is discussed in the next section.) performed during the learning process. It shouldn't be too narrow as disambiguation expansion point cannot be reached for patterns with long condition. It shouldn't be too wide either as the system will be too space-consuming. Our determination for 7 bidirectional surrounding graphemes as the width of the context is based on the result of our early investigation stating that phonetic mapping of a grapheme in any ordinary Indonesian words is ambiguous until at most 5 steps to right and/or left side. We gave 2 extra steps to anticipate extraordinary materials in the real dataset.

IG-Tree Construction. The IG-tree is a compression format of context-sensitive rules. Each path in the decision tree represents a rule and is started by a node representing a focus grapheme to be mapped to a phoneme; and the consecutive node represents the consecutive context. Information gain (IG), a computational metric based on information theory, is used to determine the order of context's expansion. Higher information gain of an attribute theoretically reflects less randomness or impurity of partitions resulted by partitioning on the attribute, while in our case it indicates more importance of the attribute in disambiguating the grapheme-to-phoneme mapping. The result of the information gain computation on our IndoG2P dataset (in its letter-by-letter format) for each attribute is described in the graphic in Fig.2.

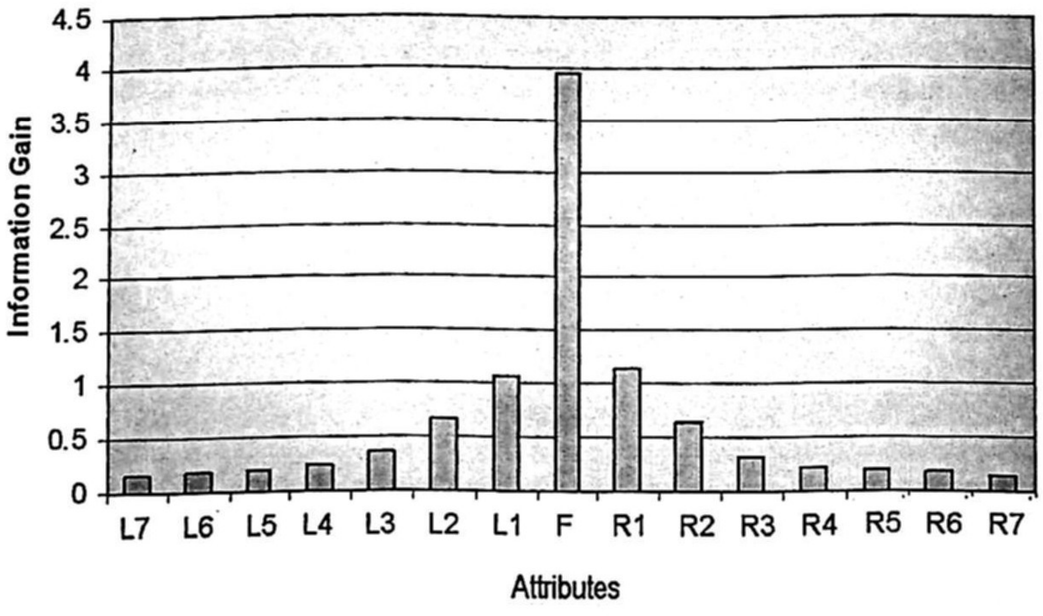

Fig. 2. Information gain for attributes in IndoG2P dataset.

Note that an attribute with the highest importance in disambiguating the grapheme-to-phoneme mapping is the attribute F (the focus grapheme). This fact is consistent with what we have stated above that the focus grapheme is located as the starting node on every path in the IG-tree. Furthermore, the graphic clearly shows us that the information gain is getting smaller as the context position is getting further from the focus grapheme. The system sorts the attributes based on their information gain values, records the order, and uses it to determine the next attribute to which the context inspection should expand during the learning process. In our case, based on the result of our computation, we got this attribute order: F – R1 – L1 – L2 – R2 – L3 – R3 – L4 – R4 – R5 – L5 – L6 – R6 – L7 – R7.

The construction of IG-tree, just similar with those of some standard decision tree structures such as ID3, C4.5, and CART, can be practically done in such a top-down divide-and-conquer manner. Performed on our letter-by-letter training set, the basic algorithm of the decision tree construction is greedy and can be expressed recursively as follows.

Base: If the current node is pure, i.e. instances reaching that node are totally of the same class, return the node as a leaf node labeled with that class. Stop developing that part of the tree;
Recurrence: Otherwise, i.e. instances reaching the current node differ in class labels, use a splitting attribute to divide the instances in such way that it splits up the example set into subsets, one for each value of the attribute. Apply this procedure to each subset.

We must add information to this recursive algorithm specifically for our case that the splitting attribute, mentioned in the recurrence, at anytime is governed by the attribute order we have discussed earlier. The order is applied constantly for every path in the decision tree.

Another feature belonging to IG-tree is the statistic recording mechanism performed on every non-leaf node. This mechanism records the phonetic label's statistic of all instances reaching each node. This statistic will be employed to perform best-guess strategy and pruning mechanism.

To illustrate how the model exactly is, we will use dataset shown in Table 2 as our study case. As the dataset is still in word-by-word version, our workflow demands it to be transformed to its letter-by-letter format. The IG-tree construction using the procedure we have discussed above is then performed on the later format of dataset. Fig. 3 illustrates a part, i.e. the part of grapheme <e> mapping paths, of the IG-tree constructed.

Retrieving Phonetic Label in IG-Tree. Given a focus grapheme and its context graphemes, the phonetic label of the focus grapheme is basically retrieved by tracing through a proper path in the IG-tree in direction to its leaf and returning the label of the leaf as the phonetic label requested. How we get phoneme /ə/ as the phonetic label of grapheme <e> in word <tempayan> will be more clearly discussed in this section. Based on the attribute order we have discussed above, the tracing will start with node labeled <e> as the focus grapheme. The node labeled with the first grapheme on the right of <e> in <tempayan>, i.e. <m>, is the second node accessed in the path. The next taken node is that labeled <t>, the first grapheme on the left of <e>; continued with node labeled <-> as the second "grapheme" on the left of <e>. It is the end of the tracing when node labeled <p>, the second grapheme on the right of <e>, is accessed, since this node is a leaf node. Thus, label /ə/ is retrieved as the phonetic label corresponding with grapheme <e> in word <tempayan>. In the similar way, phoneme /e/ can be retrieved as the phonetic label for grapheme <e> in word <tembaklah>. However, in this case our tracing will fail in reaching any leaf node when our word is, for instance, <teman> with focus on grapheme <e>. Note that the tracing gets stuck on node labeled <-> since this node have no child labeled <a>. Such problem is in our research solved with best-guess strategy.

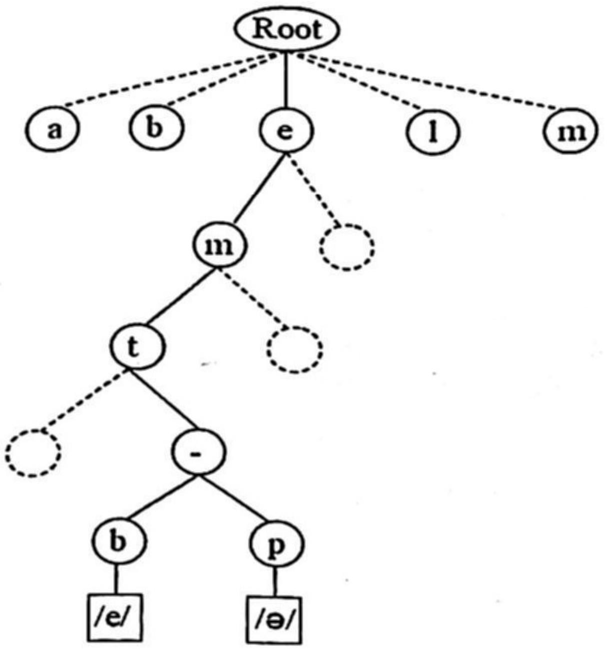

Fig. 3. The IG-tree constructed on dataset in Table 2 with stressing in <e> mapping

Best-Guess Strategy. This is a strategy employed by the system to avoid stumped mapping and to increase its generalization ability. The strategy is performed when a tracing to retrieve a phonetic label cannot reach any leaf node. When the tracing gets stuck on a node, the system will "guess" the phonetic label with the most probable label on that node. The most probable label is computed using statistic information stored on that node. The most frequent phonetic label on records affiliated with the node will be returned as the "guessed" label. In case that there are more-than-one phonetic labels are majorities, a random selection among the labels is performed.

2.4 Pruning Mechanism

When a decision tree is built, many of its branches reflect anomalies due to outliers in the learning materials. The pruning mechanism is proposed to address this problem of overfitting the data. This is performed after the IG-tree is initially grown to its entirety. Pruning is done by replacing a subtree with a new leaf node labeled with the subtree's majority phonetic label. The new tree is then validated using validation set. If the pruning step decreases the generalization accuracy, the previous subtree will be retained; otherwise, the subtree will be permanently removed by the new leaf node. In the case that the accuracy is constant, it was stated that for the same performance the more concise model is the better. This procedure is then performed on all subtrees in a bottom up fashion.

2.5 Centroid-based Text Categorization

Indonesian has some homograph words as shown in Table 5. Daelemans et al in [7] said that their research fails in handling such words in their G2P conversion system. We redeem their failure by proposing text categorization approach using the centroid-based classifier to cope with the problem.

This approach of text categorization and the main system of the IndoG2P conversion using IG-tree actually work on different level. While the IG-tree works on the level of letter with context inspection mechanism internally in its *containing* word, the approach of text categorization works on the level of text with computation on some aspects of its *contained* words / terms. It implies that this proposed approach is applicable on the system with text inputs, not only word inputs.

How centroid-based text categorization copes with the homograph problem is shortly explained as follows. In this approach each homograph word is treated as a particular problem to solve. It demands that a particular learning step, surely with a particular dataset, is performed for each homograph word. Furthermore in this approach, the text on each instance is represented as a vector in term space with TF-IDF computation for each of its dimension. A centroid model of each category is then constructed as average representation of all instances labeled with that category. When an unlabeled instance is given, the classifier computes similarity between vector representing the instance and vector representing centroid of each category. The category with the most similar centroid is output as the label for the new instance. Thus, with the centroid models constructed in the learning process, IndoG2P can correctly map the ambiguous grapheme in a given homograph word surrounded by a particular text, to its phonetic representation.

Table 5. The list of some Indonesian homograph words.

Homograph word	Ambiguous grapheme	Phonetic representation
<apel>	<e>	/apəl/,/apel/
<penanya>	<e>	/pənaña/,/penaña/
<memerah>, <pemerah>, <pemerahan>	<e>	/məmerah/, /pəmerah/, /pəmeraħan/, /məmərah/, /pəmərah/, /pəməraħan/
<seri>	<e>	/səri/,/seri/
<semi>	<e>	/səmi/,/semi/
<sedan>	<e>	/sedan/,/sədan/
<mental>	<e>	/mental, /məntal/
<seret>	<e>	/seret/, /sərət/
<serak>	<e>	/sərak/, /serak/
<tahu>	<h>	/taħu/, /tahu/
<gulai>	<i>	/gulay/,/gulai/

3 Evaluation and Results

Although both modules are practically not detachable, the IG-tree construction and the homograph handling substantially address different level of problem. Hence, their datasets are different as discussed before. So, we divided this section into two parts, each for a particular module.

3.1 IG-Tree Construction

We will see in this part the model improvements due to pruning. The performance of the final model is then evaluated using two accuracy metrics; those are accuracy-per-phoneme and accuracy-per-word. Since on the lowest level the mapping is done letter-per-letter, the computation for accuracy-per-phoneme is done to accommodate evaluation on this level. On the other hand, accuracy-per-word is more interpretable in the sense that the communication using linguistic tools in the real world is word-based, not letter-based. Technically note that accuracy-per-word must be less than or equal to accuracy-per-phoneme since to get a word correctly-mapped, its contained letters are needed to be all true, but in the opposite, one mispronounced letter is sufficient to lead the word in which it occurs to be false.

The resulted IG-tree has 2,112 leaf nodes. The number of the leaves in this section trivially represents the model's dimension. The pruning mechanism is then performed on the model to decrease the model's dimension and increase the accuracy as shown in Table 6. After 554 iterations, dimension of the final IG-tree is 57% smaller than that of the original and its accuracy for validation set is better.

Table 6. Pruning and its improvements in dimensional and accuracy for validation set.

Pruning iteration	Number of leaves	Phoneme accuracy	Word accuracy	Mean accuracy	Mean error
0	2,112	99.19	93.96	96.57	3.43
1	2,111	99.19	93.96	96.57	3.43
2	2,096	99.19	93.96	96.57	3.43
3	2,074	99.19	93.96	96.57	3.43
110	1,871	99.23	94.26	96.74	3.26
221	1,658	99.23	94.26	96.74	3.26
332	1,496	99.30	94.70	97.00	3.00
443	1,315	99.36	95.14	97.25	2.75
554	908	99.38	95.29	97.33	2.67

The final IG-tree was then tested using our test set. The test gave us result of 99.01% for phoneme accuracy and 92.42 % for word accuracy. Unfortunately, we did not find any similar system working on the same language to compare with. However, this result is much better than those of similar researches on other languages published in [3], [4], [5], [6], [8], [9], [10], and [11]. It seems that the method we used and the language we worked on are factors contributing the most for this result. It is stated in [7] that the high performance of IG-tree in G2P conversion suggests overstatement on previous knowledge-based approaches as well as more computationally expensive learning approaches. Moreover, in this research we improved the original method with new features which increase its performance. About the language we worked on, it is clear that linguistically Indonesian has much simpler phonetic rules than other languages, like English, Dutch, and French. These simple Indonesian linguistic patterns seem to be easily caught during the learning process, so the system could perform better.

Another aspect of IG-tree we want to stress on is its aspect of interpretability. As it was constructed in decision tree structure whose characteristic is descriptive as well as predictive, IG-tree gives us reason for each of its prediction. The model "teaches" us the complete detailed "lesson" about how to pronounce letters in Indonesian words. In practical level, the comprehensive rules exposed in the model constructed even perhaps can help Indonesian linguists codifying Indonesian pronunciation standards.

3.2 Homograph Handler

In this part of research we conducted experiments on five cases for five homograph words as mentioned in Table 4. However, with small number of training instances and test instances, in every single case we got so surprisingly perfect accuracy of the model built during its learning process that extremely no mistake was made by its centroid-based classifier in predicting the category of some new homograph-containing texts in its particular test set.

The well-designed features of the centroid-based classifiers are factors playing the main role in the perfect performances of the models in the five cases. The other factor is the well-prepared dataset used both in each learning and test process. In spite of their small sizes, the datasets are noise-free, discriminating, and balance.

4 Conclusion

The high performance of the system implies that the IG-tree + best-guess strategy is a powerful, language-independent, reasoning-based method well-designed to cope with grapheme-to-phoneme conversion problem nowadays. The result is furthermore contributed also by the characteristic of Indonesian itself whose pronunciation rules are relatively easy for the learning system to catch. The proposed pruning mechanism successfully improves the system's performance by significantly reducing the dimension of the model and increasing its generalization ability. The high interpretability of the model developed in this work must be considered as one aspect of its high performance as well. In addition, the homograph problems, totally unsolved or unsatisfying-solved in previous works, can be handled very well with the proposed centroid-based classifiers.

Acknowledgments

We would like to thank to Mrs. Dyas Puspandari as an Indonesian linguist and all colleagues at IT Telkom for the kind and useful suggestions.

References

1. Alwi, Hasan et al. 2003. *Tata Bahasa Baku Bahasa Indonesia*. Edisi Ketiga. Jakarta: Balai Pustaka
2. Asian, J., Williams, H. E., Tahaghoghi, S. M. M.: Stemming Indonesian. In: Proceedings of the Twenty-eighth Australasian conference on Computer Science, Newcastle, Australia, pp. 307--314 (2005)
3. Bouma, G.: A finite state and data-oriented method for grapheme-to-phoneme conversion. In: Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference, Seattle, Washington, pp. 303-310 (2000)
4. Caseiro, D., Trancoso, I., Oliveira, L., Viana, C.: Grapheme-to-phone using finite-state transducers. In: Proc. IEEE Workshop on Speech Synthesis, Santa Monica, CA, USA (2002)
5. Daelemans, W., Bosch., A.: Tabtalk: Reusability in data-oriented grapheme-to-phoneme conversion. In: In Proceedings of Eurospeech (1993)
6. Bosch, A., Daelemans, W.: Data-oriented methods for grapheme-to-phoneme conversion. In: Proceedings of the sixth conference on European chapter of the Association for Computational Linguistics, Utrecht, The Netherlands (1993)
7. Daelemans, W., Bosch, A.: Language-independent data-oriented grapheme-to-phoneme conversion. In Progress in Speech Synthesis, J. P. van Santen, R. W. Sproat, J. P. Olive, and J. Hirschberg, Eds. Springer-Verlag (1997)
8. Han, E-H., Karypis, G.: Centroid-based document classification: analysis and experimental results. In: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 424-431 (2000)
9. Reichel, Uwe D. and Florian Schiel. Using morphology and phoneme history to improve grapheme-to-phoneme conversion. In: Proceedings of the InterSpeech, pp. 1937-1940 (2005)
10. Taylor, Paul. Hidden Markov Models for grapheme to phoneme conversion. In: Proceedings of the InterSpeech, pp. 1973-1976 (2005)
11. Yvon, Francois. Self-learning techniques for grapheme-to-phoneme conversion. In: Proceeding of the 2nd Onomastica Research Colloquium, London (1994)