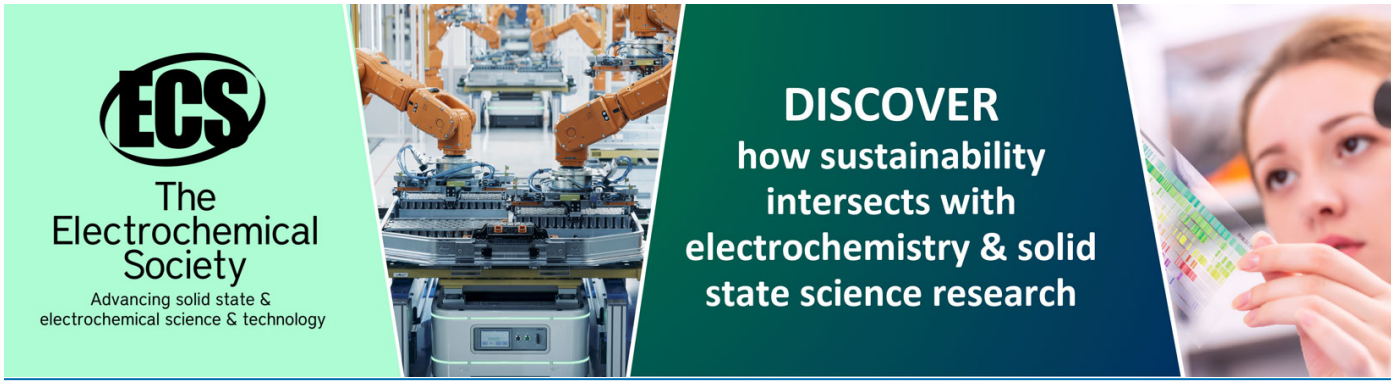
Word Embeddings Evaluation on Indonesian Translation of Al-Quran and Hadiths



Word Embeddings Evaluation on Indonesian Translation of Al-Quran and Hadiths

Muhammad Zidny Naf'an[1,2], Yunita Sari[1], and Yohanes Suyanto[1]

[1]Department of Computer Science and Electronics, Universitas Gadjah Mada, Yogyakarta, Indonesia
[2]Department of Informatics, Institut Teknologi Telkom Purwokerto, Central Java, Indonesia

E-mail: [2]muhammadzidny@mail.ugm.ac.id

Abstract. Word vectors are an important part of machine learning. Word vectors are a numerical representation of text data. One of the methods that can be used to convert text into numerics is word embeddings. The word embeddings algorithm that researchers often use is Continuous Bag of Word, Skip-Gram, and FastText. This paper will discuss the transformation of textual data from Islamic knowledge domain documents into numerical forms using these three algorithms, then evaluate the word vector results using intrinsic and extrinsic evaluation techniques. We conduct intrinsic evaluations by determining the words to be evaluated, then checking for the existence of synonyms, antonyms, related words, and derived words from the nearest set of words based on vector values. We also tried to use vector words to solve word analogy problems. The best word vector in extrinsic evaluation is the result of the CBOW algorithm which is integrated with Binary Relevance and Multilayer Perceptron, with an accuracy value of 77.56% and a hamming loss value of 8.14%.

1. Introduction

Word embeddings are the vectors that represent words as a point in a multidimensional semantic space [1]. Some words that have semantic relations (synonym, antonym, word similarity, word relatedness, etc.) will have close vectors. Word embeddings will avoid the sparse matrix form in word representation, as happens when TF-IDF is used to represent a word [2]. That is, word embeddings will produce dense vectors with the predefined size of dimensions. The dense vector is also more effective while used as input features in the machine learning algorithm and prevent overfitting conditions. Dense vectors are also better in capturing synonyms than sparse vectors [1].

Word embeddings have been widely used in natural language processing and text mining, such as text similarity [3], text classification [4,5], multi-label text classification [6,7], sentiment analysis [8], machine translations [9,10], etc. Moreover, previous research works have applied word embedding for domain-specific text analysis (i.e. social science [11], religion [12], health and medical [13,14], mining [15], and cybersecurity [16]).

Based on searches, there are currently no pre-trained word embeddings for Islamic domain-based documents. In addition, there is also no research on evaluating word embeddings from documents based on the Islamic domain. Evaluation on word embeddings is quite important. Because word embeddings can affect the performance of machine learning algorithms. We hope that this word

embeddings evaluation research can be used as a reference in choosing the right word embeddings algorithm for Islamic domain text.

In this paper, we extract word embeddings from Indonesian text (i.e. Al-Quran translation, Al-Quran interpretation, and hadith translation) using the Continuous Bag of Word (CBOW), Skip-Gram, and FastText algorithms. Word embeddings will be evaluated using intrinsic and extrinsic evaluation methods. Intrinsic evaluation is the word embeddings quality measures based on experiments in which word embeddings are compared to human judgments on word relations (synonym, antonym, etc.). Extrinsic evaluation is an evaluation technique that measures word embeddings quality based on the performance of the machine learning algorithm or other NLP tasks with word embeddings as the feature vectors. The main objective of this paper is to find the best word embeddings model from textual data of Islamic domain documents in the Indonesian language.

The main contributions of this paper are:

- developing word embedding Indonesian Quran and Hadiths Translation,
- provide a thorough evaluation on the developed word embeddings.

2. Literature Review

Nooralahzadeh, et. al. [15] evaluated domain-specific word embeddings from oil and gas textual data. About 8 million texts were trained on CBOW and Skip-Gram algorithm with a hyperparameter tuning experiment. They conducted two evaluations approach, which are intrinsic and extrinsic evaluations. On intrinsic evaluation, they randomly chose 100 unique words on the oil and gas domain then retrieved the 10-most-similar words for each unique word provided in the word embeddings. They evaluated word embeddings based on synonym, antonymy, and alternative form from the 10-most-similar words to evaluate the tuning experiment. After getting the best parameter, they analyzed the error of the 10-most-similar words using several categories: spelling variant, alternative form, references-synonym, human-judge synonym, antonym, hypernym, hyponym, co-hyponym, holonym, meronym, related, unrelated/unknown. The frequent errors occur in related categories, hyponym, and co-hyponym. On extrinsic evaluation, word embeddings were used as the features of the multi-label classification task. They ran basic CNN by Kim (2014) and several modified CNN models for the experiment. The best result obtained was from CNN models with two embedding layers (randomly embedding vector and domain-specific vector) integrated with the retrofitting method and Out of Vocabulary (OOV) handling.

Similar to Nooralahzadeh, et. al. work's, Sarma, et. al. [14] trained "Substances User Disorders" (SUDs) dataset with a modified word embeddings algorithm for resulting high-quality word embeddings vector. They combined large scale corpora embeddings and domain-specific embeddings using linear Canonical Correlation Analysis (CCA) or a nonlinear kernel CCA (KCCA). They found that the CCA/KCCA with combined word embeddings improves substantially over the generic embeddings. They evaluated combined word embeddings in extrinsic evaluation tasks on binary sentiment classification with the Logistic Regressor method. These architectures tested on four public datasets: Yelp, Amazon, IMDB, and A-Chess.

Another work by Roy, et. al. on [16] evaluated their proposed method on two cybersecurity text corpora: a malware description corpus and a "Common Vulnerability and Exposure (CVE)" corpus. The authors developed a novel Annotation and Word Embedding (AWE) algorithm. They stated that there was a diversification of the types of domain knowledge. To overcome this, the authors built the protection of text annotations in the form of predicate structural arguments. The basis of AWE is the Word2Vec system with input to the AWE algorithm is the text and annotations that will produce output in the form of vector representations of words and annotations.

3. Methodology

We define the steps of this research in this section. Then we describe the technical implementation steps of the algorithms CBOW, Skip-gram, and FastText. We use the Word2Vec and FastText libraries in the Python programming language.

3.1. Methodology

The research methodology of this study can be seen in Fig. 1. We began by collecting text data from the Indonesian translation of Al-Quran, the Indonesian interpretation *(tafseer)* of the Al-Quran, and the Indonesian translation of the hadith in the book of Sahih Al-Bukhary.

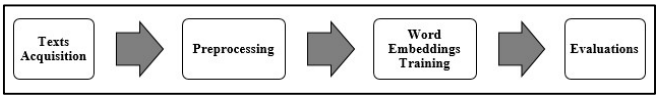


Figure 1. The research methodology

After collecting text data, we performed pre-processing the text, namely: case folding, sentence parsing, removing numbers and punctuations, repeated words checking (example: “buku-buku” *(books)*), and words tokenisation. From sentence parsing part, we obtained 157,870 sentences with 45,327 unique words. In Bahasa Indonesia, plural words are formed by repeating nouns and adding dashes between the two words [17], for example, the word "buku-buku" which means "many books". Therefore, it is necessary to check for repeated words so that the meaning of the words does not change and they are not split into two tokens.

Specifically, for the hadith text, we applied the Named Entity Recognition (NER) process to recognize the names of narrators. We use rule-based NER based on word writing patterns. Words that indicate the name of the narrator are written in square brackets ([...]). This process is intended so that the series of narrators names are not separated when doing the tokenization process. Example of original text and NER result on Table 1.

Table 1. Example of original text and text after NER

Original text	NER result
Telah menceritakan kepada kami **[Abdullah bin Yusuf]** berkata, telah mengabarkan kepada kami **[Malik]** dari **[Hisyam bin 'Urwah]** dari **[bapaknya]** dari **[Aisyah]**……."	Telah menceritakan kepada kami **abdullah_bin_yusuf** berkata, telah mengabarkan kepada kami **malik** dari **hisyam_bin_urwah** dari bapaknya dari **aisyah** ……."

3.2. Implementation

There are two global architectures in Word2Vec [18,19], those are CBOW and Skip-Gram. We implemented both of them and FastText algorithm [20], then choose the best one.

Mikolov, et. al. [19] create Word2Vec to reduce computational complexity on the Neural Network Language Model (NNLM), developed by Bengio, et. al (2013). On NNLM, There is a probability distribution calculation on the hidden layer for all words in the vocabulary (V as the vocabulary set and |V| as the number of words in the dictionary) and produces an output with a V dimension length. So Mikolov et. al. used hierarchical softmax, where the vocabulary is represented as a Huffman binary tree. With this model Mikolov et. al. can minimize computing costs [19]. Mikolov tries two architectures to build word vectors, that are Continous Bag-of-Word (CBOW) and Continous Skip-Gram. The difference between these two architectures is seen in the purpose of architecture. The CBOW predicts the target word (w_i) based on the context (surrounding word from w_i), and the Skip-gram aims to predict the surrounding words given the word w_i.

Joulin, et. al. built FastText architecture for extracting word embeddings from texts [5]. FastText is similar to the CBOW model, where the middle word is replaced by a label, and they use a bag of n-grams as features to get some partial information from the local word.

First, we build the architecture of word embeddings, with default values for hyper-parameters, i.e. *dim* = 100, *win* = 5, *min_count* = 5, and *neg* = 5, while *dim* is the dimensionality of vector or vector size, *win* is the size of windows for grabbing context words, *min_count* is the minimum frequency of

the word, and *neg* is negative sampling size. Gensim library [21] was used for implementing CBOW, Skip-Gram (SG), and FastText architectures. Based on Table 2, it can be seen that the CBOW algorithm has the fastest training time than the other algorithms.

Table 2. Time consuming when building bag of word and model (without phrase). The value of window = 5, size = 100, and min_count = 5

Architecture Name	Time for Building BOW (seconds)	Time for Building Model (seconds)
CBOW	4.00	846.91
Skip-Gram (SG)	3.82	2414.06
FastText	5.52	3542.96

4. Evaluation

Based on our best knowledge, there is still no gold standard resource available to evaluate the semantic distribution of words in Bahasa as they already exist for English, such as Simlex-999 [22], SimVerb-3500 [23], etc. Therefore, we built our domain-specific gold standard in the field of Islam based on the glossary available in Islamic religious education books [24]. We chose 75 unique words from [24].

In this evaluation, we used an evaluation model defined by Bakarov [25] in which he divided the word embeddings evaluations into two major parts, namely extrinsic and intrinsic evaluation. Extrinsic evaluation methods are based on the ability to use word embeddings as the feature vectors of supervised machine learning algorithms or as used in one of NLP tasks. The performance of the supervised model as a measure of the quality of word embeddings.

Table 3. Examples of words and its related words from CBOW algorithm. Syn = synonym, Ant = antonym, Rel = Related Word, Der = Derivative, and Conc = Conclusion

Word	Most Related Word	Syn	Ant	Rel	Der	Conc
zhalim	aniaya	1	0	0	0	1
zhalim	adillah	0	1	0	0	1
zhalim	sewenang-wenang	1	0	0	0	1
zhalim	semenamena	1	0	0	0	1
zhalim	zalim	1	0	0	0	1
zhalim	kejam	1	0	0	0	1
zhalim	berwenang	0	0	0	0	0
zhalim	lalim	1	0	0	0	1
zhalim	curang	1	0	0	0	1
zhalim	adil	0	1	0	0	1

4.1. Intrinsic Evaluation

The intrinsic evaluation method will evaluate word embeddings based on human judgment (expert judgment). In this evaluation, we used human evaluators who would evaluate interrelated words produced by the word embedding architecture.

On [15], three kinds of word relation for evaluating word embedding were used, i.e. synonym, antonym, and alternative form. We replaced alternative forms to related and derivative forms. Furthermore, we used Indonesian Thesaurus by Kateglo.com for synonym (ex. *adil* → *jujur*), antonym (*lemah* → *kuat*), a related word (*nabi* → *rasul*), and derivative form (*iman* → *keimanan*).

For each word in the glossary (supposed word *g*), we extract 10 related words (words w_n) and group them into one of these following categories: synonym, antonym, related words, or derived words. If there is one category that matches, between *g* and w_n then this pair is given the conclusion

value of 1, otherwise 0. Table 3 shows the example of 10 related words from “zhalim” (harsh) and the category of each related word.

We use Equation (1) to calculate true presentation for each word embeddings algorithm.

$$\frac{1}{N}\sum_{i=1}^{N} \max(Syn_i , Ant_i , Rel_i , Der_i) \quad (1)$$

Where $Syn_i , Ant_i , Rel_i , Der_i \in \{0,1\}$, and N is the number of rows.

Table 4 describes the results of the evaluation of each algorithm. The algorithm with the best results is Skip-Gram and FastText that managed to get 52.53% of the most related words that can be categorized.

Table 4. Result of human judgment intrinsic evaluation

Model	True Presentation For Each Row	Error
CBOW	51.73%	48.27%
Skip-Gram	52.53%	47.47%
FastText	52.53%	47.47%

4.2. Word Analogy

The second most popular intrinsic evaluation is the word analogy method (some researchers also calls it linguistic regularities, analogical reasoning, or word semantic coherence) [25]. The idea of word analogy is solving mathematical operation with the operands are word vectors. For example, given a pair of words ***a*** and ***a****, and the third word is ***b***. Then the relation based on the analogy between ***a**** with ***a*** can be used to predict ***b**** from ***b*** which has the same relationship ***a**** with ***a***. For example, *car*:*wheel* :: *bird*:*b**, meaning that the *wheel* is "a part of" the *car*, so *b** is “part of” the *bird* (wings, beaks, claws, etc.) [26].

To get the prediction of the word ***b**** it is necessary to know the vector value of ***b****. Mathematically, to get the vector ***b****, it can be done with a simple vector operation as described in Equation (2). Furthermore, based on vector ***b****, we can search for the word with the closest vector to vector ***b**** as shown in Equation (2) [27].

$$w_{b*} \approx w_{a*} - w_a + w_b \quad (2)$$

In the study of English linguistics, it is easier to find a public dataset as the ground truth of analogy issues. Gao, et. al. [28] proposed WordRep dataset which is divided into 26 semantic classes, WordRep was developed from the "Google Analogy" data by Mikolov, et. al. [19] which combines morphological and semantic elements in pairing words. However, it is different for Bahasa Indonesia. The resources supporting Indonesian language studies are still very rare. Likewise, for the analogy dataset, based on our research no one has yet made an analogy dataset for Indonesian. Therefore, in testing with this analogy approach, we created our own dataset. The dataset we made are consisted of 38 pairs which contained 6 categories, namely antonyms, synonyms, hyponyms, hypernym, derivative words, and related words.

Based on the words analogy example, given a pair of words ***a***, ***a****, and ***b*** we determine the word ***b**** by applying Equation (2) and looking for words in vocabulary that have a similar vector to vector ***b****. The determination of similar words uses the method "3CosMul" [29] from Gensim library as described in Equation (3).

$$\underset{b*\in V}{\operatorname{argmax}} \frac{\cos(b*,b)\ \cos(b*,a*)}{\cos(b*,a)+\varepsilon} \quad (3)$$

with V is vocabularies and ε = 0.001 use for preventing division by zero.

The results of the analogy test are shown in Table 5. Overall, the three embedding algorithms can solve analogy problems, especially in the class of synonyms, antonyms, derivative forms, and related words. Based on Table 5, it can be seen that the best algorithm that is able to solve word analogy problems is Skip-Gram. However, it failed when completing the hyponym and hypernym classes.

Because hypernym and hyponym or vice versa are rarely appeared together as contexts and target words. In the derivative class, the FastText algorithm has 100% accuracy, because when FastText conducts training on words, this algorithm takes subwords based on *n-grams* word. This makes FastText succeed in retrieving similar words from morphologically similar words. For example on pair of words:

budaya:*kebudayaan* :: *bangsa*:*b** (cultural:culture :: nation:b*).

The ***a*** is a basic word, and ***a**** is a word ***a*** that is affixed with "ke-an". So ***b**** should be "kebangsaan". FastText obtained top-5 prospective answers as follows: “bangsa-bangsa” (nations), “berbangsa-bangsa” (nations), “kebangsaan” (nationality), “sebangsa” (countrymen), “mangsa” (prey).

Table 5. Evaluation result using analogy problem on CBOW architecture

Class	CBOW	Skip-Gram	FastText
antonym	60.00%	60.00%	60.00%
derivative	60.00%	80.00%	100.00%
hypernym	40.00%	40.00%	0.00%
hyponym	40.00%	80.00%	40.00%
related	46.20%	61.50%	30.80%
synonym	100.00%	100.00%	100.00%
Average	57.70%	**70.25%**	55.13%

4.3. Extrinsic Evaluation

The next evaluation uses an extrinsic evaluation approach, we used word vectors from the CBOW algorithm, Skip-Gram, and FastText as the feature of multilabel text classification. For this experiment, we use the dataset from [30]. The contents of this dataset are the Indonesian translation text of Hadiths Shahih Bukhary. This dataset consists of 1064 rows where each row has 3 labels: [suggestion, prohibition, information] and each label represented in binary value {0, 1}. We split dataset into three parts: 45% for training, 22% for validation, and 33% for testing.

We implemented pre-processing steps to the text including case folding, word normalization, remove punctuation, and tokenizing.

Table 6. The best parameter of learning methods

Word Embedding	Learning Method	Validation Accuracy	Best Parameters
CBOW	BR+MLP	78,09%	alpha = 1e-05 hidden layer = (100, 10)
CBOW	BR+SVM	75,00%	kernel = RBF
Skip-Gram	BR+MLP	78,37%	alpha = 1e-06 hidden layer = (100,)
Skip-Gram	BR+SVM	75,00%	kernel = RBF
FastText	BR+MLP	78,23%	alpha = 1e-06 hidden layer = (100, 10)
FastText	BR+SVM	75,00%	kernel = RBF

In this evaluation, we solve the Multilabel Classification (MLC) task using the Binary Relevance (BR) problem transformation approach. In BR, we combined two different learning algorithms, Multi-Layer Perceptron (MLP) and Support Vector Machine (SVM). BR will treat each label as a separate single-label classification problem. We implemented the classification algorithms using Scikit-Multilearn library [31].

We performed hyperparameter tuning for MLP with this following parameter values: *alpha* = {0.00001, 0.000001} and *hidden_layer_sizes* = {(100,), (100.10,), (100,10,3,)}. While for the SVM, we tried three different kernels which are Radial Base Function (RBF), sigmoid, and linear. The learning algorithm input is an average of word vectors for each document.

Table 6 shows the optimal parameter for each scenario. After getting the optimal parameters, then we re-train the learning algorithm using the optimal parameters. We used accuracy score, hamming loss, f1-micro average score, and f1-macro average score as the evaluation metrics.

Hamming loss calculates the symmetric difference between the predicted label and the ground truth then calculate the number of mismatches label set. The F1-macro-averaging evaluates each label by summing the number of true positives, false positives, true negatives and false negatives, and independently computes the F1-measure for each label. While the F1-micro-averaging counts once after the F1-measures for all labels have been collected [32].

Based on Table 7, the average testing accuracy of Multilabel Classification (MLC) is 76% and the average F1-macro average can reach 89%. This result proves that word vectors produced in this study are promising. Based on testing accuracy and hamming loss values, the best word vectors on extrinsic evaluation is the result of the CBOW algorithm integrated with Binary Relevance and Multilayer Perceptron.

Table 7. Result of multi-label classification using the best parameter

Word Embedding	Learning Method	Testing Accuracy	Hamming Loss	F1-micro average	F1-macro average
CBOW	BR+MLP	77.56%	8.14%	91.00%	77.00%
CBOW	BR+SVM	71.88%	10.42%	88.00%	73.00%
Skip-Gram	BR+MLP	77.56%	8.33%	91.00%	77.00%
Skip-Gram	BR+SVM	75.28%	9.75%	88.00%	56.00%
FastText	BR+MLP	77.56%	8.52%	90.00%	78.00%
FastText	BR+SVM	75.28%	9.75%	88.00%	56.00%

5. Conclusion

This paper has discussed intrinsic and extrinsic evaluations of the corpus with the specific domain of religious texts. Based on our research results, CBOW is the fastest algorithm when conducting training on the data we use. Meanwhile, based on the results of the intrinsic test, the best algorithms in categorizing related words are Skip-Gram and CBOW with an accuracy rate of 52.53%. We also tested word vectors extracted from the text to solve word analogy problems, where the best performance was achieved by the Skip-Gram algorithm with an average accuracy rate of 70.25%.

Apart from intrinsic testing, we also perform extrinsic testing. In this type of test, word vectors become feature learning on the Multilabel Classification (MLC) task for hadith text. The best algorithm for this MLC task is CBOW which is combined with Binary Relevance and Multilayer Perceptron (MLP). The level of accuracy obtained by CBOW+BR+MLP is 77.56% with a hamming loss value of 8.14%.

From this study, we conclude that special pre-processing (such as NER) are needed for certain data. For example, in the hadith text, it is necessary to separate the narrators and contents of hadith *(matn)* parts. For further research, we plan to perform fine-tuning in the word embeddings architecture so that the expected results of the word vector become more optimal.

References

[1] Jurafsky D and Martin J H 2008 *Speech and Language Processing* (New Jersey: Prentice Hall)
[2] Chen G, Ye D, Xing Z, Chen J and Cambria E 2017 Ensemble application of convolutional and recurrent neural networks for multi-label text categorization *2017 Int. Jt. Conf. Neural Networks* pp 2377–83
[3] vor der Brück T and Pouly M 2019 Text similarity estimation based on word embeddings and

matrix norms for targeted marketing *Proc. of the 2019 Conf. of the North* (Stroudsburg, PA, USA, Minnesota: Association for Computational Linguistics) pp 1827–36
[4] Kim Y 2014 Convolutional neural networks for sentence classification *Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar) pp 1746–51
[5] Joulin A, Grave E, Bojanowski P and Mikolov T 2017 Bag of tricks for efficient text classification *Proc. of the 15th Conf. of the European Chapter of the Association for Computational Linguistics* vol 2 (Melbourne:Association for Computational Linguistics) pp 427–31
[6] Georgakopoulos S V., Vrahatis A G, Tasoulis S K and Plagianakos V P 2018 Convolutional neural networks for toxic comment classification *Proc. of the 10th Hellenic Conf. on Artificial Intelligence* (Patras, Greece: Association for Computing Machinery)
[7] Okky Ibrohim M, Sazany E and Budi I 2019 Identify abusive and offensive language in Indonesian Twitter using deep learning approach *J. of Physics: Conf. Series* vol **1196** (Bristol: IOP Publishing)
[8] Petrolito R and Dell'Orletta F 2016 Word embeddings in sentiment analysis *Proc. of the Fifth Italian Conf. on Computational Linguistics CLiC-it 2018* (Torino: Accademia University Press)
[9] Cholakov K and Kordoni V 2016 Using word embeddings for improving statistical machine translation of phrasal verbs *Proc. of the 12th Workshop on Multiword Expressions* (Berlin, Germany: Association for Computational Linguistics) pp 56–60
[10] Zheng Z and Chen J 2018 Modeling past and future for neural machine translation *Trans. Assoc. Comput. Linguist.* **6** 145–57
[11] Kozlowski A C and Taddy M 2019 The Geometry of culture : analyzing the meanings of class through word embeddings *Am. Sociol. Rev.* **84** pp 905–49
[12] Alturayeif N S 2017 *Text Mining and Similarity Measures of the Quran and the Bible* (Leeds: University of Leeds)
[13] Jiang Z, Li L, Huang D and Jin L 2015 Training word embeddings for deep learning in biomedical text mining tasks *2015 IEEE Int. Conf. Bioinforma. Biomed.*
[14] Sarma P K, Liang Y and Sethares W A 2018 Domain adapted word embeddings for improved sentiment classification *56th Annual Meeting of the Association for Computational Linguistics (Short Papers)* (Melbourne: Association for Computational Linguistics) pp 37–42
[15] Nooralahzadeh F, Øvrelid L and Lønning J T 2018 Evaluation of domain-specific word embeddings using knowledge resources *Proc. of the Eleventh Int. Conf. on Language Resources and Evaluation (LREC-2018)* pp 1438–45
[16] Roy A, Park Y and Pan Sh 2017 Learning domain-specific word embeddings from sparse cybersecurity texts *Comput. Res. Repos.* **abs/1709.0**
[17] Tim Pengembang Pedoman Bahasa Indonesia Badan 2016 *Pedoman Umum Ejaan Bahasa Indonesia* (Jakarta, Indonesia: Badan Pengembangan dan Pembinaan Bahasa Kementerian Pendidikan dan Kebudayaan)
[18] Mikolov T, Sutskever I, Chen K, Corrado G and Dean J 2013 Distributed representations of words and phrases and their compositionality *NIPS*
[19] Mikolov T, Chen K, Corrado G and Dean J 2013 Efficient estimation of word representations in vector space *CoRR* **abs/1301.3**
[20] Bojanowski P, Grave E, Joulin A and Mikolov T 2017 Enriching word vectors with subword information *Trans. Assoc. Comput. Linguist.* **5** pp 135–46
[21] Řehůřek R and Sojka P 2010 Software Framework for topic modelling with large corpora *Proc. of LREC 2010 workshop New Challenges for NLP Frameworks* (Valletta, Malta: University of Malta) pp 46–50
[22] Hill F, Reichart R and Korhonen A 2015 SimLex-999: Evaluating semantic models with (genuine) similarity estimation *Comput. Linguist.* **41** 665–95

[23] Gerz D, Vulic I, Hill F, Reichart R and Korhonen A 2016 Simverb-3500: a large-scale evaluation set of verb similarity *EMNLP 2016 - Conf. on Empirical Methods in Natural Language Processing, Proc.* (Austin, Texas: Association for Computational Linguistics) pp 2173–82
[24] Marzuki 2012 *Pembinaan Karakter Mahasiswa Melalui Pendidikan Agama Islam di Perguruan Tinggi Umum* (Yogyakarta: Pusat Mata Kuliah Universiter, Universitas Negeri Yogyakarta)
[25] Bakarov A 2018 A survey of word embeddings evaluation methods *Preprint arXiv:1801.09536*
[26] Wang B, Wang A, Chen F, Wang Y and Kuo C-C J 2019 Evaluating word embedding models: methods and experimental results *APSIPA Trans. Signal Inf. Process.* **8** pp 1–13
[27] Allen C and Hospedales T 2019 Analogies Explained: towards understanding word embeddings *Preprint arXiv:1901.09813*
[28] Gao B, Bian J and Liu T-Y 2014 WordRep: a benchmark for research on learning word representations *Preprint arXiv:1407.1640*
[29] Levy O and Goldberg Y 2014 Linguistic regularities in sparse and explicit word representations *Proc. of the Eighteenth Conf. on Computational Language Learning* (Baltimore, Maryland, USA) pp 171–80
[30] Bakar M Y A, Adiwijaya A and Al Faraby S 2018 Multi-label topic classification of hadith of Bukhari (Indonesian Language Translation) using information gain and backpropagation neural network *Proc. of the 2018 Int. Conf. on Asian Language Processing, IALP 2018* (IEEE) pp 344–50
[31] Szymański P and Kajdanowicz T 2019 Scikit-multilearn : a scikit-based Python environment for performing multi-label classification *J. Mach. Learn. Res.* **20** pp 1–22
[32] Charte F, Rivera A J, Charte D, del Jesus M J and Herrera F 2018 Tips, guidelines and tools for managing multi-label datasets: The mldr.datasets R package and the Cometa data repository *Neurocomputing* **289** pp 68–85