Jurnal Pseudocode, Volume VI Nomor 2, September 2019, ISSN 2355-5920
www.ejournal.unib.ac.id/index.php/pseudocode

PENGELOMPOKAN ARTEFAK DOKUMEN PERANGKAT LUNAK *OPEN SOURCE* DENGAN VEKTOR PARAGRAF

Guntur Budi Herwanto

Ilmu Komputer/ Departemen Ilmu Komputer dan Elektronika,
Fakultas Matematik dan Ilmu Pengetahuan Alam,
Universitas Gadjah Mada
Yogyakarta

[1]gunturbudi@ugm.ac.id

Abstrak: Dalam beberapa tahun belakangan, perangkat lunak *open source* semakin bertumbuh. Tidak hanya perangkat lunak dalam bentuk final, namun komponen dan *library* perangkat lunak semakin berkembang setiap tahunnya. Github merupakan salah satu lokasi populer dalam mempublikasikan *project open source*. Ketersediaan *dataset* yang besar ini merupakan peluang bagi peneliti di bidang perangkat lunak *development* dalam mengembangkan risetnya. Perkembangan variasi artefak perangkat lunak membuat metode yang bersifat supervised menjadi sulit. Penilitian ini mencoba untuk melakukan pengelompokkan secara unsupervised dengan teknik *clustering K*-Means dan representasi paragraph vector. Langkah ini merupakan awalan dalam pembentukan model klasifikasi yang membutuhkan supervisi dalam pelabelan dokumennya. Hasil *clustering* menunjukkan dokumen dapat dapat di kelompokkan menjadi beberapa *cluster* dan hasil yang terbaik dilihat pada cluster dengan *k* berjumlah 6.
Kata Kunci: *document clustering*, doc2vec, *k-means clustering*, artefak perangkat lunak.

Abstract: In recent years, open source software is growing. Not only is software in its final form, but components and software libraries are growing every year. Github is one of the popular locations for publishing open source projects. The availability of this large dataset is an opportunity for researchers in the field of software development in developing their research. The development of variations in software artifacts makes supervised methods difficult. This research tries to do unsupervised grouping with K-Means clustering technique and paragraph vector representation. This step is a start in the formation of a classification model that requires supervision in labeling documents. Clustering results show documents can be grouped into clusters and the best results are seen in clusters with k equal 6.
Keywords: document clustering, latent dirichlet allocation, ward hierarchical clustering, topic modelling

I. PENDAHULUAN

Dalam beberapa tahun belakangan, *open source* perangkat lunak semakin bertumbuh. Tidak hanya perangkat lunak dalam bentuk final, namun komponen dan library perangkat lunak semakin berkembang setiap tahunnya. Github [1] merupakan salah satu lokasi populer dalam mempublikasikan *project open source*, karena sifatnya yang gratis dan memiliki komunitas yang besar. Oleh karena itu, banyak perangkat lunak developer baik dari yang sifatnya kecil sampai perusahaan besar, mempublikasikan *project open source*nya dalam Github. Ketersediaan dataset yang besar ini merupakan peluang bagi peneliti di bidang perangkat lunak development dalam mengembangkan risetnya. Namun data yang sangat besar dan heterogen membuat peneliti menjadi kesulitan mendapatkan informasi yang tepat. Lebih jauh lagi sangat banyak data yang bentuknya teks bebas dan tidak terstrukur.

Dalam penelitian [2] telah dibahas mengenai jenis-jenis perangkat lunak artifacts yaitu *source code*, *application*, *archive*, *audio*, *disk image*, *font*, *image*, *project file*, *testing code*, dan file lainnya. Selain itu file lainnya yang penting adalah file dokumentasi. File dokumentasi ini biasanya dapat berupa *design documentation*, *List of Contributors*, *Requirement Documents*, Contributors Guide, *License*, *Release Notes*, dan *Setup Files*. Dalam penelitian sebelumnya ekstensi file sebuah artefak perangkat lunak dapat menjadi panduan dalam pengelompokan jenis artefak. Namun untuk file yang berupa teks bebas seperti dokumentasi, akan cukup sulit dalam mengelompokkan karena mempunya ekstensi yang tidak spesifik seperti doc, txt, ataupun pdf.

Perkembangan variasi artefak perangkat lunak membuat metode yang bersifat *supervised* menjadi sulit. Penilitian ini mencoba untuk melakukan pengelompokkan secara *unsupervised* dengan teknik clustering K-Means dan representasi dokumen doc2vec. Langkah ini merupakan awalan dalam pembentukan model klasifikasi yang membutuhkan supervisi dalam pelabelan dokumennya.

II. TINJAUAN PUSTAKA

Text mining atau *text analytics* telah digunakan sebagai alat untuk mengekstrak informasi dalam artefak perangkat lunak seperti yang dilakukan oleh [3] dalam mengekstrak informasi mengenai arsitektur perangkat lunak. Selain dokumentasi perangkat lunak, dapat ditemukan bahwa ekstraksi dokumen yang memiliki pengetahuan mengenai arsitektur adalah tujuan yang paling penting [3].

Sebagian besar penelitian terutama berfokus pada perangkat lunak *open source* atau *dataset* yang terbuka [2], [4]–[6]. Peneliti rekayasa perangkat lunak telah dapat mengambil manfaat dari dataset *open source* seperti Github untuk melakukan subjek penelitian potensial [1] dengan bantuan alat [7]. Peneliti dapat dengan mudah mengambil dataset besar dari alat ini. Namun, kumpulan data ini bisa sangat besar, jika kita mencoba melihatnya secara berurutan. Dengan demikian, pendekatan otomatis diperlukan untuk memahami file-file dasar yang dihasilkan pada proyek-proyek *open source* [2].

Representasi teks ke dalam bentuk vektor merupakan langkah pertama sebelum kemudian dilakukan *clustering*. Langkah ini juga dapat dibilang menjadi langkah yang sangat penting dan banyak teknik yang diteliti untuk dapat membuat bentuk yang terbaik untuk menunjukkan representasi dari kumpulan teks. *Bag-of-words* (BOW) merupakan teknik yang sangat terkenal dan cukup mudah untuk diimplementasikan.

Namun, karena sifatnya yang sangat sintaktik, dan tidak memperhatikan urutan dari kata, maka hasil dari pemorsesan selanjutnya tidak terlalu baik.

Representasi lain yang cukup baik juga adalah *Term Frequency – Inverse Document Frequency* (TF-IDF). Penggunaannya hampir sama dengan BOW hanya saja TF-IDF mempertimbangkan *Document Frequenc*y. Jika sebuah kata muncul di banyak dokumen, maka kata tersebut dapat dianggap sebagai kata yang tidak penting. Sebaliknya jika sebuah kata jarang muncul di dokumen-dokumen, dan banyak muncul di sebuah dokumen, maka kata tersebut penting untuk dokumen tersebut. Intuisi ini terbukti baik dalam beberapa kasus pemrosesan teks. Namun TF-IDF juga tidak memperhatikan urutan maupun konteks semantik dalam kalimat.

Latent Dirichlet Allocation (LDA) [8] juga merupakan teknik umum untuk pemodelan topik (mengekstraksi topik/ kata kunci dari teks). Representasi topik juga dapat digunakan dalam merepresentasikan fitur dari sebuah kata. Namun hasil dari LDA sangat sulit untuk dievaluasi performanya.

III. METODE

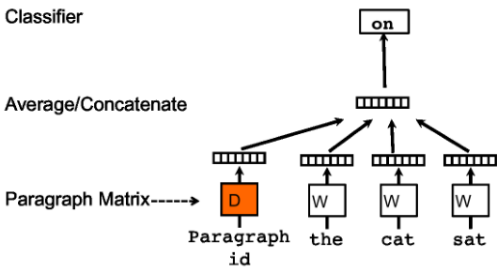
Dalam bagian ini akan dijelaskan metode yang digunakan dalam penelitian ini. Terdapat dua langkah besar yaitu representasi dokumen dengan menggunakan metode doc2vec dan clustering dengan menggunakan K-Means.

3.1 Representasi Teks ke dalam Vektor

Tahap ini merupakan tahapan yang krusial dalam menentukan *cluster* yang baik. Jika gagal merepresentasikan teks ke dalam vektor secara baik, maka pengelompokan cluster bisa jadi tidak sesuai. Metode klasik seperti *bag-of-words* kurang baik dalam menangkap intuisi semantik serta urutan kata dalam dokumen. Oleh karena itu terdapat metode yang dapat menangkap intuisi semantik dari kata yaitu word2vec [9]. Metode ini terbukti baik dalam mereprentasikan semantik dari setiap kata.

Dalam perkembangannya, Le dan Mikolov [10] mengusulkan *Paragraph Vector*, teknik *unsupervised* yang mempelajari representasi vektor kontinyu dan terdistribusi untuk potongan teks. Teks yang dimaksud disini panjangnya dapat bervariasi, mulai dari kalimat hingga dokumen. Nama *Paragraph Vector* adalah untuk menekankan fakta bahwa metode ini dapat diterapkan pada potongan teks yang panjangnya bervariasi, mulai dari frasa atau kalimat hingga dokumen besar. Teknik ini dinilai cocok untuk merepresentasikan dokumen yang pada umumnya memiliki ukuran yang besar.

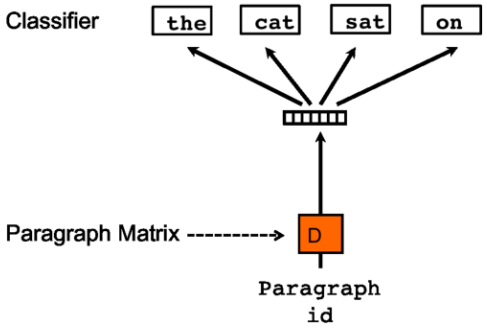
Terdapat dua variasi teknik yang dapat diterapan untuk *paragraph vector*. Gambar 1 merupakan ide dasar dari *parapraph vector*.Tiga kata *the*, *cat*, *sat*, digunakan untuk memprediksi kata keempat yaitu *on*. Elemen yang paling kiri yaitu paragraf id, dan berupa matriks mewakili informasi yang hilang dari konteks saat ini dan dapat bertindak sebagai memori topik paragraf. Model ini dinamakan *Paragraph Vector Distributed Memory* dan model ini dapat dilihat pada Gambar 1.



Gambar 1. Paragraph Vector Distributed Memory (PV-DM) [10]

Model kedua dari vektor paragraf mengabaikan konteks kata dalam input, tetapi membuat model

untuk memprediksi kata-kata yang diambil secara acak dari paragraf di output. Model ini dinamakan *Paragraph Vector Distributed Bag of Words* dan model ini dapat dilihat pada Gambar 2.



Gambar 2. Paragraph Vector Distributed Bag of Words [10]

3.2 *Clustering* dengan K-Means

Algoritme *k-means* digunakan untuk mempartisi sekumpulan data yang diberikan ke jumlah cluster *k* yang telah ditentukan. Algoritme seperti yang dijelaskan oleh [11] dimulai dengan random-set *k centroid* (μ). Selama setiap langkah pembaruan, semua pengamatan x di-*assign* ke *centroid* terdekat. Hal ini dijelaskan dalam formula (1). Dalam algoritme k-means yang standar, satu poin hanya merujuk ke satu *centroid*. Jika terdapat beberapa centroid yang memiliki jarak yang sama, maka akan dipilih secara acak.

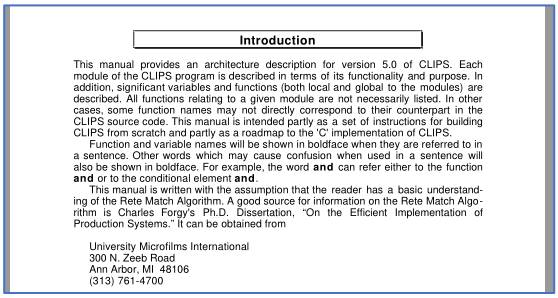
$$S_i^{(t)} = \{x_p : |x_p - \mu_i^{(t)}|^2 \leq |x_p - \mu_j^{(t)}|^2 \; \forall j, 1 \leq j \leq k\} \quad (1)$$

Setelah itu, *centroid* dilakukan pembaharuan dengan menghitung rata-rata pengamatan yang ditugaskan ke masing-masing *centroid*. Perhitungan dapat dilihat pada formula (2).

$$\mu_i^{(t+1)} = \frac{1}{\left|S_i^{(t)}\right|} \sum_{x_j \in S_i^{(t)}} x_j \quad (2)$$

IV. HASIL DAN PEMBAHASAN

Bagian ini akan dijelaskan mengenai hasil dan pembahasan penelitian yang telah dilakukan. Dataset yang digunakan berasal dari file *open source* github. Dalam penelitian ini kami hanya berfokus pada data teks bebas yang ber ekstensi pdf, doc, ataupun ppt. Dataset yang berhasil didapatkan adalah 3.686 artefak perangkat lunak. Gambar 3 menunjukkan contoh dari artefak perangkat lunak yang berupa dokumen arsitektur.



Introduction

This manual provides an architecture description for version 5.0 of CLIPS. Each module of the CLIPS program is described in terms of its functionality and purpose. In addition, significant variables and functions (both local and global to the modules) are described. All functions relating to a given module are not necessarily listed. In other cases, some function names may not directly correspond to their counterpart in the CLIPS source code. This manual is intended partly as a set of instructions for building CLIPS from scratch and partly as a roadmap to the 'C' implementation of CLIPS.

Function and variable names will be shown in boldface when they are referred to in a sentence. Other words which may cause confusion when used in a sentence will also be shown in boldface. For example, the word **and** can refer either to the function **and** or to the conditional element **and**.

This manual is written with the assumption that the reader has a basic understanding of the Rete Match Algorithm. A good source for information on the Rete Match Algorithm is Charles Forgy's Ph.D. Dissertation, "On the Efficient Implementation of Production Systems." It can be obtained from

University Microfilms International
300 N. Zeeb Road
Ann Arbor, MI 48106
(313) 761-4700

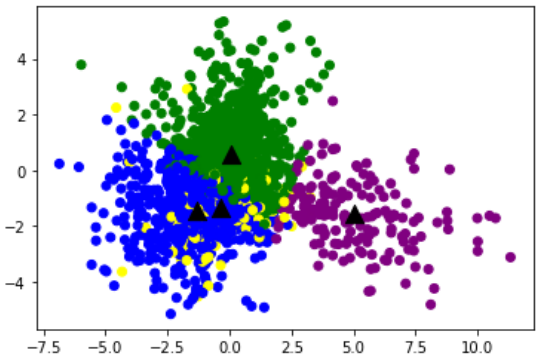
Gambar 3. Contoh artefak perangkat lunak

Seluruh dokumen kemudian direpresentasikan ke dalam bentuk doc2vec dan dilakukan *clustering k-means* dengan 3 variasi *k* yaitu 4, 6, dan 8 cluster. Performa dari clustering dapat akan dinilai dari beberapa *metrics* [12] yaitu *Adjusted rand score*, *Adjusted mutual info score*, *Homogeneity score*, *Completeness Score*, *V-measure score* [13], dan *Silhouette score*. Hasil lengkap dapat dilihat pada Tabel 1.

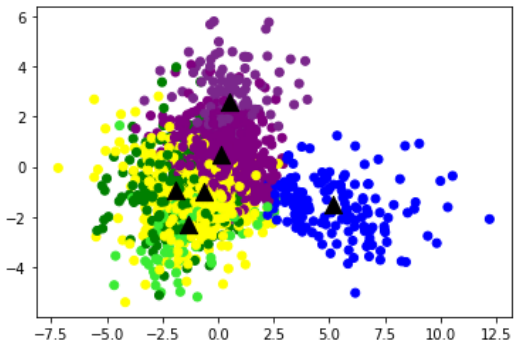
Tabel 1. Hasil Clustering dengan K-means

Parameter	**k=4**	**k=6**	**k=8**
Homogeneity score	0.921	0.929	0.746
Completeness score	0.887	0.928	0.723
V measure score	0.904	0.929	0.734
Adjusted rand score	0.951	0.963	0.809
Adjusted mutual info score	0.904	0.928	0.733
Silhouette score	0.268	0.256	0.209

Visualisasi *clustering* dengan memanfaatkan *principal component analysis* dapat dilihat pada Gambar 4 dan Gambar 5.



Gambar 4. Hasil K-Means dengan 4 *cluster*



Gambar 5. Hasil K-Means dengan 6 *cluster*

Dalam hasil *cluster* dapat terlihat bahwasannya k=6 secara umum menghasilkan *cluster* yang terbaik.

V. KESIMPULAN

Penelitian ini merupakan awalan dari proses labeling yang selanjutnya akan dilakukan untuk menghasilkan *classifier* artefak dokumen perangkat lunak yang lebih baik. Pengelompokan akan lebih memudahkan pelabel dalam menentukan jenis artefak. Dari hasil yang didapatkan, kluster yang terbaik didapatkan ketika K berjumlah 6. Jika dilihat secara manual dari artefak dokumen perangkat lunak, variasi ini sesuai dengan penelitian sebelumnya yang membagi tipe artefak dokumen menjadi 7 tipe.

DAFTAR PUSTAKA

[1] N. Munaiah, S. Kroh, C. Cabrey, and M. Nagappan, "Curating GitHub for engineered software projects," *Empir. Softw. Eng.*, vol. 22, no. 6, pp. 3219–3253, 2017.

[2] Y. Ma, S. Fakhoury, M. Christensen, and V. Arnaoudova, "Automatic Classification of Software Artifacts in Open-Source Applications," in *The Mining Software Repositories (MSR)*, 2018.

[3] T. Bi, P. Liang, A. Tang, and C. Yang, "A systematic mapping study on text analysis techniques in software architecture," *J. Syst. Softw.*, vol. 144, no. May, pp. 533–558, 2018.

[4] M. Soliman, M. Galster, and M. Riebisch, "Developing an Ontology for Architecture Knowledge from Developer Communities," *Proc. - 2017 IEEE Int. Conf. Softw. Archit. ICSA 2017*, pp. 89–92, 2017.

[5] W. Ding, P. Liang, A. Tang, H. Van Vliet, and M. Shahin, "How do open source communities document software architecture: An exploratory survey," *Proc. IEEE Int. Conf. Eng. Complex Comput. Syst. ICECCS*, pp. 136–145, 2014.

[6] G. Robles, J. M. Gonzalez-barahona, J. L. Prieto, U. Rey, and J. Carlos, "Assessing and Evaluating Documentation in Libre Software Projects ∗," *Hum. Factors*, no. 004337, 2006.

[7] G. Gousios, "The {GHT}orrent dataset and tool suite," in *Proceedings of the 10th Working Conference on Mining Software Repositories*, 2013, pp. 233–236.

[8] D. Blei, L. Carin, and D. Dunson, "Probabilistic topic models," *IEEE Signal Process. Mag.*, vol. 27, no. 6, pp. 55–65, 2010.

[9] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, 2013, pp. 3111–3119.

[10] Q. Le, T. Mikolov, and T. G. Com, "Distributed Representations of Sentences and Documents," vol. 32, 2014.

[11] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. Fifth Berkeley Symp. Math. Stat. Probab. Vol. 1 Stat.*, 1967, pp. 281–297.

[12] N. X. Vinh, "Information Theoretic Measures for Clusterings Comparison : Variants , Properties , Normalization and Correction for Chance," vol. 11, pp. 2837–2854, 2010.

[13] A. Rosenberg and J. Hirschberg, "{V}-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning ({EMNLP}-{C}o{NLL})*, 2007, pp. 410–420.