BAB II

STUDI LITERATUR

2.1. Kesamaan Dokumen & Plagiarisme

Sebelum kita mengetahui lebih lanjut terkait metode yang akan digunakan, wajib diketahui definisi dan perbedaan dari Kesamaan Dokumen (*Document Similarity*) dan Plagiarisme agar penelitian ini tetap sejalan dengan konteks yang dibicarakan.

2.1.1. Definisi Kesamaan Dokumen

Menurut kamus *Oxford*, *similarity* merupakan *"The state or fact of being similar."* [6] sementara similar sendiri memiliki arti *"Having a resemblance in appearance, character, or quantity, without being identical."* [7]. Sehingga dapat dikatakan bahwa kesamaan dokumen ini memiliki arti keadaan dimana suatu dokumen memiliki kemiripan dengan dokumen lain baik dalam segi tampilan, karakter atau makna meskipun tidak identik.

2.1.2. Definisi Plagiarisme

Dalam dunia akademik, kesamaan dokumen menjadi sangat penting dan seringkali dikaitkan dengan tindak plagiarisme. Menurut Peraturan Menteri Pendidikan RI No. 7 Tahun 2010, "Plagiat adalah perbuatan sengaja atau tidak sengaja dalam memperoleh atau mencoba memperoleh kredit atau nilai untuk suatu karya ilmiah, dengan mengutip sebagian atau seluruh karya dan atau karya ilmiah pihak lain yang diakui sebagai karya ilmiahnya, tanpa menyatakan sumber secara

tepat dan memadai" [8]. Sehingga dapat kita lihat dengan mengutip sebagian atau seluruh karya dan atau karya ilmiah pihak lain, karya ilmiah tersebut memiliki kesamaan dokumen dengan dokumen yang dikutipnya.

2.1.3. Kesamaan Dokumen dengan Plagiarisme

Menurut H. Soelistyo dalam bukunya yang berjudul Plagiarisme: Pelanggaran Hak Cipta dan Etika [9], Plagiarisme dapat dibagi kedalam beberapa tipe yang dimana dari keseluruhan tipe tersebut pasti adanya kemiripan suatu dokumen dengan dokumen lainnya. Namun jika suatu kemiripan tersebut memiliki teknik sitasi yang baik dan benar, maka dokumen tersebut tidak dapat dinyatakan sebagai plagiasi.

Sehingga dapat ditarik kesimpulan bahwa kesamaan dokumen belum tentu plagiarisme sementara plagiarisme sudah pasti mengandung kesamaan dokumen.

2.2. Text Mining

Dalam mendeteksi kesamaan dokumen dengan dokumen-dokumen lain, terdapat beberapa metode yang bisa kita implementasikan. Pada dasarnya, metode-metode tersebut dapat disebut dengan metode text-mining.

Dalam terjemahan bahasa Indonesia, memang text mining berarti penambangan teks. Namun representasi dari text mining sendiri dapat didefinisikan sebagai proses pengetahuan intensif di mana pengguna berinteraksi dengan kumpulan dokumen dari waktu ke waktu dengan menggunakan seperangkat alat analisis [10]. Dengan cara yang analog dengan data mining, text mining berupaya mengekstraksi informasi yang berguna dari sumber data melalui identifikasi dan

eksplorasi pola yang menarik. Namun dalam kasus text mining, sumber data adalah koleksi yang terdokumentasi, dan pola yang menarik tidak ditemukan di antara catatan basis data yang diformalkan tetapi dalam data tekstual yang tidak terstruktur dalam dokumen dalam koleksi ini.

2.2.1. Preprocessing

Sebelum dapat diolah, awalnya teks yang akan digunakan dalam text mining yang begitu acak dan tidak teratur / tidak konsisten diproses dengan beberapa ragam cara sehingga selanjutnya teks tersebut dapat dengan mudah diolah. Langkah preprocessing biasanya terdiri dari tugas-tugas seperti *tokenization, filtering, lemmatization* dan *stemming* [10]. Berikut ini dijelaskan secara singkat proses-proses tersebut:

1. ***Tokenization***

   Tokenisasi adalah tugas memecah urutan karakter menjadi beberapa bagian (kata / frasa) yang disebut token, dan mungkin pada saat yang sama membuang karakter tertentu seperti tanda baca. Daftar token kemudian digunakan untuk diproses lebih lanjut.

2. ***Filtering***

   Pemfilteran biasanya dilakukan pada dokumen untuk menghapus beberapa kata. Pemfilteran umum adalah penghapusan kata-berhenti. Berhenti kata adalah kata-kata yang sering muncul dalam teks tanpa memiliki banyak informasi konten (mis. Preposisi, kata sambung, dll). Demikian pula kata-kata yang sering muncul dalam teks membantu untuk memiliki sedikit informasi untuk membedakan dokumen yang berbeda dan juga kata-kata

yang muncul sangat jarang juga mungkin tidak memiliki relevansi yang signifikan dan dapat dihapus dari dokumen.

3. Lemmatization

Lemmatization adalah tugas yang mempertimbangkan analisis morfologis dari kata-kata, yaitu mengelompokkan berbagai bentuk kata yang terefleksi sehingga dapat dianalisis sebagai satu item. Dengan kata lain, metode lemmatization mencoba memetakan bentuk kata kerja menjadi infinite tense dan kata benda ke bentuk tunggal. Untuk lemmatize dokumen pertama-tama kita harus menentukan POS dari setiap kata dokumen dan karena POS membosankan dan rentan kesalahan, dalam praktiknya metode stemming lebih disukai.

4. Stemming

Metode stemming bertujuan untuk memperoleh stem (root) dari kata-kata yang diturunkan. Proses ini menguraikan bentuk dari suatu kata menjadi bentuk kata dasar. Secara singkat, dalam proses ini kata-kata berimbuhan akan didirubah menjadi kata dasar. Algoritma stemming memang tergantung pada bahasa yang digunakan.

2.2.2. Algoritma Text Mining & Document Similarity

Dalam dunia text mining, berbagai penelitian telah dilakukan. Algoritma yang ada pun sangatlah beragam. Namun yang difokuskan dalam tulisan ini adalah algoritma-algoritma Semantic Textual Similarity (STS). Terdapat 2 algoritma paling dasar yang harus kita pelajari sebelum melihat pada algoritma-algoritma tersebut yaitu *Vector Space Model* dan *Cosine Similarity*.

1. *Vector Space Model*

Vector Space Model (VSM) adalah model untuk merepresentasikan dokumen dalam bentuk aljabar. VSM adalah standar yang diterima secara luas yang digunakan setiap kali representasi numerik teks diperlukan [11]. Ide dasar VSM adalah untuk merepresentasikan teks sebagai Bag of Words (BoW). Untuk memiliki representasi yang ringkas, urutan kata dalam dokumen diabaikan dan dokumen tersebut diwakili oleh vektor frekuensi kata. Secara lebih formal, kosa kata dibuat di mana setiap kata atau istilah memiliki indeks integer unik. Dokumen diwakili oleh vektor kolom di mana setiap elemen menyimpan frekuensi kata dalam dokumen.

2. *Cosine Similarity*

Cosine similarity merupakan metode yang digunakan untuk menghitug tingkat kesamaan (similarity) antar dua buah objek. Untuk tujuan klastering dokumen, fungsi yang baik adalah fungsi cosine similarity [12]. Untuk notasi himpunan dapat digunakan rumus

$$Sim(X,Y) = \frac{|X \cap Y|}{|X|^{\frac{1}{2}} \, |Y|^{\frac{1}{2}}} \qquad 1)$$

Dimana $|X \cap Y|$ adalah jumlah term yang ada pada dokumen X dan dokumen Y, $|X|$ adalah jumlah term yang ada pada dokumen X dan $|Y|$ adalah jumlah term pada dokumen Y.

Dalam penelitian terdahulu, VSM dan cosine similarity terbukti kompeten dalam perbandingan dokumen dan deteksi plagiarisme dokumen thesis berbahasa Indonesia [13].

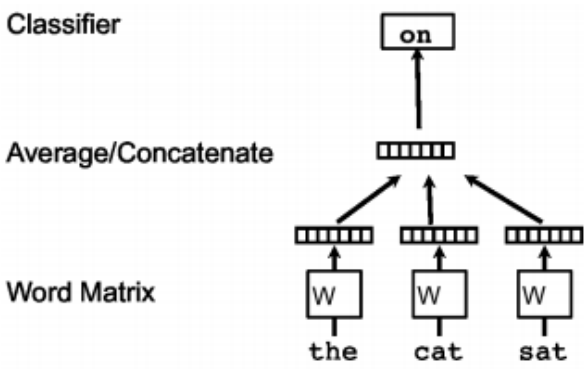
Penelitian yang telah dilakukan oleh Alvarez dan Bast yang membahas tentang state-of-the-art dari algoritma-algoritma tersebut [5] membandingkan beberapa algoritma, diantaranya adalah:

1. ***Doc2Vec***

   Paragraph Vector, yang lebih dikenal dengan *Doc2Vec* adalah penggabungan tingkat paragraf dari tim peneliti yang bertanggung jawab untuk *Word2Vec*.

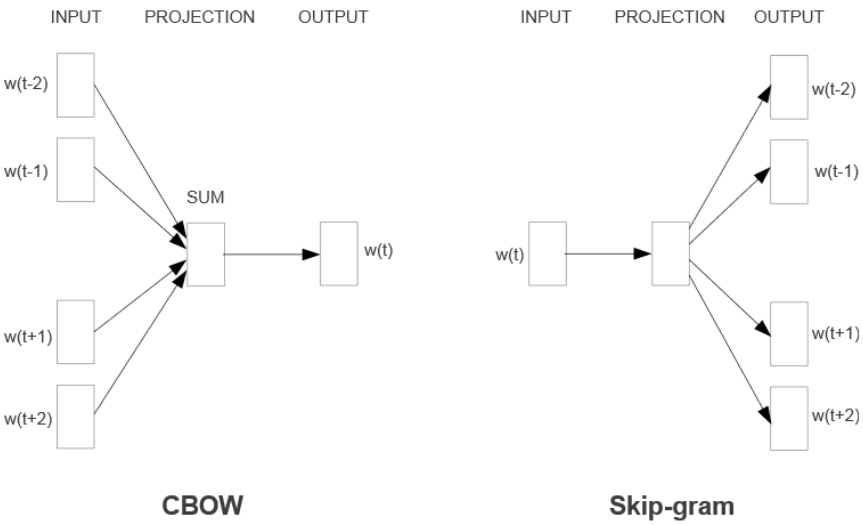
   Untuk mengenal *Doc2Vec*, pertama kita harus mengetahui tentang *Word2Vec* terlebih dahulu. Secara singkat, *Word2Vec* merupakan algoritma yang dikenal untuk merepresentasikan kata-kata dalam teks menjadi vector. Dikenalkan oleh Mikolov dan kawan-kawan, *Word2Vec* menggabungkan dua algoritma yang ada sebelumnya yaitu *Continuous bag of words* (CBOW) dan *Skip-Gram* model [14].

   CBOW membentuk semacam area bergeser disekitar kata yang ada, untuk memprediksi nya dari konteks yang ada. Setiap kata direpresentasikan sebagai vector fitur. Setelah dilakukan percobaan, vektor-vektor tersebut menjadi vector kata.



Gambar 2.1. Algoritma CBOW [2]

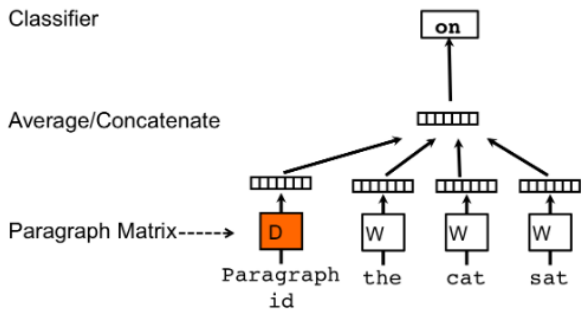
Algoritma selanjutnya adalah Skip-Gram Model. Algoritma ini berbanding terbalik dengan CBOW dimana alih-alih menggunakan sebuah kata setiap saat, digunakanlah 1 kata untuk memprediksi seluruh kata disekitarnya (konteks). Skip gram jauh lebih lamban dibandingkan dengan CBOW tetapi dipertimbangkan lebih akurat jika menggunakan kata-kata yang jarang.



Gambar 2.2. Algoritma Word2Vec [14]

Seperti yang dikatakan, tujuan *doc2vec* adalah untuk membuat representasi numerik dari suatu dokumen, terlepas dari panjangnya. Tetapi tidak seperti kata-kata, dokumen tidak datang dalam struktur logis seperti kata-kata, jadi metode lain harus ditemukan.

Konsep dari peniliti sendiri sangat sederhana namun sangat berpengaruh: mereka menggunakan model *word2vec*, dan menambahkan vektor lain (Paragraph ID di bawah). Meskipun sederhana, Doc2Vec mampu mengungguli algoritma perbandingan dokumen klasik lainnya seperti Jaro-Winkler yang cukup popular [15].



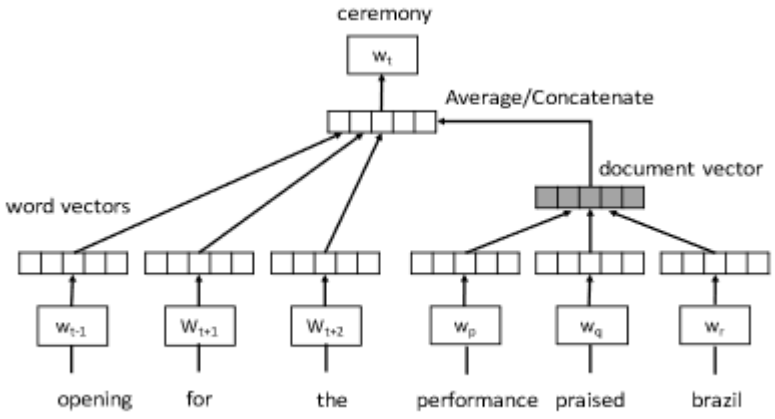
Gambar 2.3. Algoritma Doc2Vec [2]

2. *Doc2VecC*

Selanjutnya ada *Doc2VecC [3]*, merupakan pengembangan dari *Doc2Vec* yang menggunakan variable tambahan yaitu *coruption*. Meskipun tujuan utama Doc2VecC adalah untuk menghasilkan embeddings dokumen, pada kenyataannya, Doc2VecC adalah algoritma *word embedding* seperti Word2Vec. Perbedaannya adalah bahwa *word embeddings* secara khusus

dilatih sedemikian rupa sehingga rata-rata dokumen menghasilkan representasi yang bermakna dengan noise minimal.

Arsitektur Doc2VecC sangat mirip dengan Doc2Vec. Keduanya didasarkan pada model C-BoW Word2Vec, di mana area ukuran tetap digunakan untuk memindai corpus. Model belajar untuk memprediksi kata pusat di area, target, mengingat kata-kata di sekitarnya di area, konteks. Pada Doc2Vec, penyematan dokumen dilatih dengan melampirkan vektor bersama sebagai kata yang disematkan ke setiap konteks dalam dokumen. Setelah pelatihan, vektor dokumen ini akan berisi informasi global dari dokumen yang mengisi informasi yang hilang dalam konteks lokal dan membantu memprediksi kata target. Namun, Doc2Vec memiliki kelemahan serius: menyimpulkan penyematan untuk dokumen baru itu mahal karena masalah optimisasi perlu dipecahkan lagi untuk dokumen baru ini sambil menjaga embeddings tetap terkunci.



Gambar 2.4. Algoritma Doc2VecC [3]

Doc2VecC mengatasi ini dengan modifikasi sederhana. Alih-alih mempelajari vektor dokumen global ini dari awal, mereka dibuat dengan rata-rata semua kata embeddings dalam dokumen. Untuk alasan yang sama Doc2Vec menghasilkan representasi global yang baik, Doc2VecC akan melatih embeddings kata yang optimal untuk rata-rata ke makna semantik global dokumen. Ini membuat model lebih mudah untuk dilatih, karena lebih banyak informasi dibagikan. Ini juga meremehkan tugas menyimpulkan embeddings dokumen baru.

Aspek kunci lain dari Doc2VecC adalah penggunaan korupsi. Untuk mengurangi biaya komputasi, Doc2VecC secara acak mengabaikan bagian-bagian yang signifikan dari teks dan sengaja menghilangkan dimensi dari penyematan dokumen saat pelatihan. Memperkenalkan suara seperti itu mungkin tampak berlawanan dengan intuisi, tetapi itu adalah praktik umum di banyak bidang pembelajaran mesin. Ini menghindari overfitting, meningkatkan generalitas dan menciptakan variasi acak dari data pelatihan untuk secara efektif meningkatkan jumlah sampel data dan memanfaatkan dataset secara maksimal. Ini setara dengan menambahkan noise Gaussian ke regresi linier atau menggunakan drop-out dalam model saraf yang mendalam. Para penulis juga membuktikan bahwa korupsi menerapkan regularisasi implisit dengan sifat-sifat yang diinginkan, seperti mengurangi pengaruh kata-kata yang sangat sering.

3. Sent2Vec

Sent2Vec [4] merupakan algoritma sentence embedding. Pengamatan utama dalam Sent2Vec adalah bahwa ada dua tren berbeda di dunia algoritma embedding. Beberapa penulis bersikeras menerapkan arsitektur saraf yang mendalam, seperti Skip-thinking. Model-model ini memiliki kapasitas lebih untuk mempelajari pola yang rumit tetapi sangat mahal untuk dilatih. Sebaliknya, model dangkal tidak sekuat tetapi melatih mereka sangat murah sehingga mereka dapat mengambil keuntungan dari dataset pelatihan yang jauh lebih besar. Kontras yang tepat ini dicontohkan beberapa tahun yang lalu dengan penemuan Word2Vec. Ada banyak iterasi model saraf untuk embeddings kata sebelumnya, tetapi gagasan menggunakan model yang jauh lebih sederhana dan sejumlah besar korpora menjadikan Word2Vec terobosan seperti sekarang.

Sent2Vec adalah metode untuk melakukan hal yang sama untuk kalimat atau dokumen embeddings. Model seperti *Skip-thoughts* menggunakan arsitektur saraf yang mendalam dengan ribuan bobot internal untuk dilatih. Namun, publikasi terbaru telah menunjukkan bahwa, dalam banyak kasus, centroid embedding tertimbang sederhana mengungguli model yang lebih kuat ini. Sama seperti Doc2VecC, Sent2Vec memoles konsep menanamkan centroid dan mengusulkan model yang sederhana namun efisien seperti Word2Vec yang dapat menelan korpora yang jauh lebih besar dan mengalahkan yang mutakhir.

Sent2Vec, dalam praktiknya, hampir identik dengan model cBoW dari Word2Vec. Seperti pada algoritma embedding berbasis jendela lainnya, jendela konteks dipindai secara berurutan melalui corpus. Kata sentral di jendela adalah kata target dan sisanya adalah konteksnya. Dalam cBoW, kata target diprediksi berdasarkan konteksnya. Dalam prakteknya, ini berarti bahwa embedding konteks dirata-ratakan dan produk titik antara embedding target dan centroid konteks diminimalkan.

2.3. Model Fragmentasi

Dalam proses perbandingan dokumen, kita tahu bahwa suatu dokumen pastinya tidak akan sama persis 100% secara keseluruhan. Meskipun suatu dokumen dengan dokumen yang lain memiliki tujuan dan pembahasan yang sama, penulisan dan kata-kata pastinya akan memiliki perbedaan. Sesuai dengan tipe plagiarisme yang telah dijelaskan oleh H. Soelistyo [9], rata-rata tiap plagiarisme membahas suatu bagian dalam artikel baik kata, kalimat maupun gagasan yang tidak mencantumkan sitasi yang baik dan benar.

Maka dari itu dapat disimpulkan bahwa perbandingan dokumen baiknya adalah dilakukan dengan konteks yang tepat. Dalam dokumen karya sesuai buku metode penelitian [16], memiliki beberapa bagian yang memiliki konteks yang berbeda-beda. Sehingga untuk membandingkan suatu dokumen dengan yang lain, haruslah dibandingkan bagian per-bagian.

Dari sisi teknisnya, fragmentasi dapat memberikan nodes yang lebih luas [17] sehingga prediksi kata dalam algoritma perbandingan dokumen dapat lebih

efektif dan efisien. Proses fragmentasi dokumen dapat membantu terdistribusinya informasi kedalam metode atau algoritma perbandingan dokumen yang lebih baik.

2.4. Perbandingan dengan Penelitian Terdahulu

Metode-metode yang telah dibahas sebelumnya diatas merupakan beberapa metode pendukung dari salah satu metode yang akan digunakan dalam penelitian ini. Namun dalam implementasinya dijelaskan pada tabel 2.1 berikut beberapa penelitian terdahulu terkait Kesamaan Dokumen dan Plagiarisme.

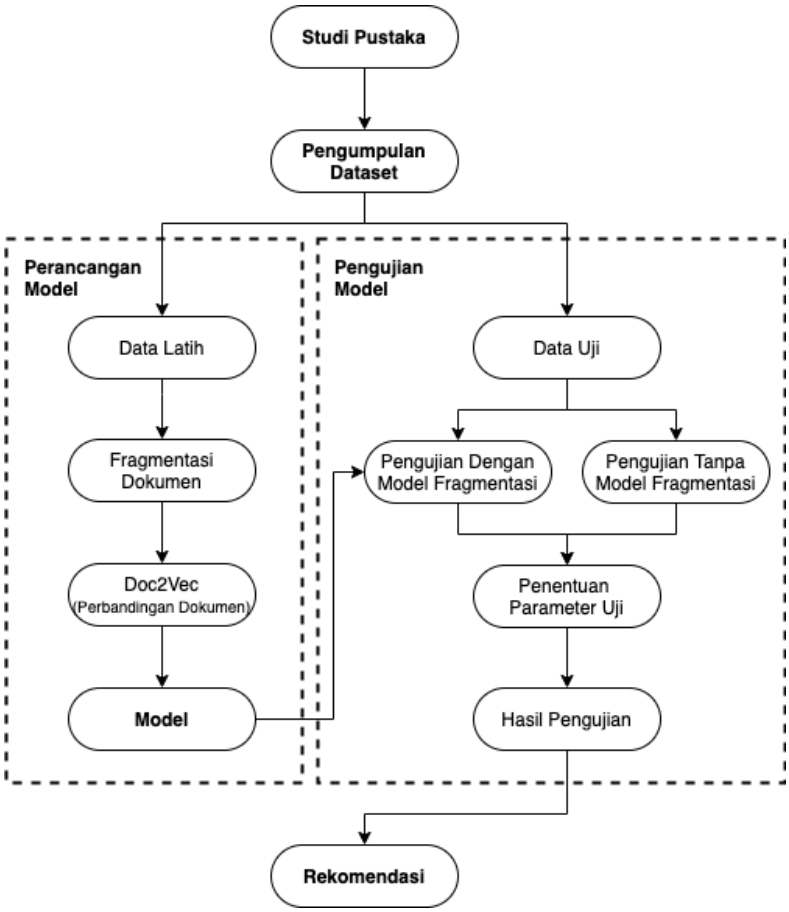
Table 2.1 Perbandingan dengan penilitian terdahulu

Judul Penelitian	Metode	Objek
Document Similarity using Dense Vector Representation [18]	Doc2Vec, Doc2VecC	Job Posting, English
Term Extraction and Document Similarity in an Integrated Learning Design Environment [19]	VSM, Cosine Similarity	Student Work, English
Plagiarism Detection for Indonesian Texts [20]	Ngrams	Text, Bahasa Indonesia
Deteksi Plagiasi Dokumen Skripsi Mahasiswa Menggunakan Metode N-Grams Dan Winnowing [21]	N-grams, Winnowing	Skripsi, Bahasa Indonesia
Klasifikasi Sentiment Analysis pada Review Film Berbahasa Inggris dengan Menggunakan Metode Doc2Vec dan Support Vector Machine (SVM) [22]	Doc2Vec, SVM	Text, Bahasa Indonesia
Model Perbandingan Dokumen Karya Ilmiah Dengan Metode Fragmentasi Menggunakan Algoritma Kesamaan Dokumen Doc2vec	**Doc2Vec, Fragmentasi**	**Karya Tulis Ilmiah, Bahasa Indonesia**

Dapat dilihat dari beberapa perbandingan pada tabel tersebut, bahwa perbedaannya adalah metode Doc2Vec belum pernah digunakan pada objek Karya Tulis Ilmiah berbahasa Indonesia.

2.5. Alur Penelitian (*Roadmap*)

Dari beberapa pustaka yang telah dikaji sebelumnya, maka disusun suatu kerangka penelitian yang menjelaskan alur dari penelitian ini. Berikut ini merupakan bagan alur atau tahapan-tahapan kegiatan yang dilakukan pada penelitian ini.



Gambar 2.6. Alur Penelitian (Roadmap)