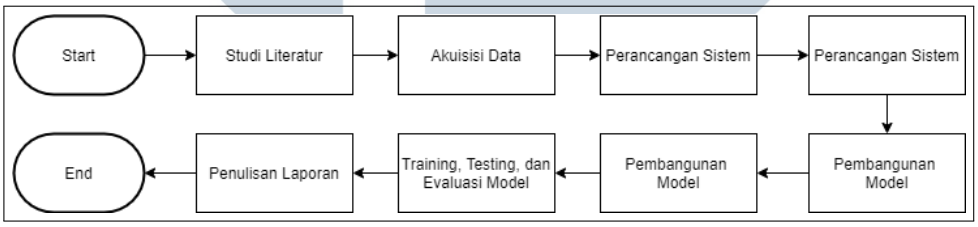
BAB 3
METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

3.1.1 Gambaran Umum

Penelitian yang dilakukan dapat dijabarkan menjadi beberapa tahapan. Penelitian dimulai dengan studi literatur sebagai landasan penelitian. Selanjutnya melakukan persiapan sebelum implementasi sistem yaitu akuisisi data dan perancangan sistem. Lalu, implementasi sistem dilakukan dengan pembangunan model, *training*, *testing*, dan *evaluasi* model dengan dataset yang sudah diperoleh sebelumnya. Penelitian diakhiri dengan penulisan laporan untuk keperluan dokumentasi penelitian yang telah selesai dilakukan. Tahapan-tahapan tersebut dapat dilihat pada gambar 3.1.



Gambar 3.1. Flowchart Gambaran Umum Metodologi Penelitan

3.1.2 Studi Literatur

Studi literatur dilakukan dengan membaca dan memahami artikel penelitian terkait seperti jurnal ilmiah, buku, dan karya tulis lainnya. Hal ini bertujuan untuk menggali informasi terkait subyek yang akan diteliti. Teori-teori terkait penilitian ini yaitu berita palsu, *text processing*, *word embedding algorithm*, dan *support vector machine*.

3.1.3 Akuisisi Data

Dataset *training* dan *testing* didapatkan dari situs web kaggle yang telah dipublikasikan di jurnal IEEE Transactions on Computational Social Systems. Dataset ini merupakan gabungan dari beberapa dataset berita yang bersumber dari empat media penyiaran berita meliputi Kaggle, BuzzFeed, McIntire, dan Reuters.

Berita yang terdapat pada dataset ini dalam bahasa inggris dan memiliki topik mengenai politik Amerika Serikat. Pemilihan bahasa dan topik ini dikarenakan dataset yang digunakan memiliki sumber yang sama dengan penelitian terdahulu [2]. Hal ini dapat memperjelas perbandingan metode dan hasil pada penelitian ini dengan penelitian terdahulu tersebut. Dataset memiliki empat kolom yaitu *Serial Number* (nomor seri sebagai index), *Title* (judul artikel berita), *Text* (isi artikel berita), dan *Label* (jenis berita asli atau palsu) [18].

3.1.4 Perancangan Sistem

Sebelum melakukan *preprocessing* dataset, hal yang harus dilakukan adalah merancang jalannya implementasi algoritma. Perancangan dilakukan dengan pembuatan *flowchart* yang menjelaskan cara kerja algoritma dalam mendeteksi berita palsu. Selanjutnya adalah tahap *preprocessing* dataset. *Preprocessing* diawali dengan menghapus entri data null pada dataset. Lalu, data akan dipisahkan menjadi data *training* dan data *testing*. Setelah itu, fitur teks berita akan dipisahkan dari nomor seri, judul, dan label.

3.1.5 Pembangunan Model

Dataset yang sudah bersih akan dilakukan *feature extraction* menggunakan *pre-trained* GloVe dari *library* Gensim. *Library* Gensim dipilih karena *library open source* yang selain menyediakan pre-traiend model, Gensim juga menyediakan *function* untuk membaca model tersebut. Gensim juga merupakan *library* yang melatih model *word-embedding* pada korpora besar seperti Wikipedia dan Google News [19].

Algoritma klasifikasi SVM akan dibangun menggunakan *library* scikit-learn. *Library* scikit-learn adalah *library open source* yang dibuat khusus untuk menyediakan berbagai *tools* yang digunakan pada pengembangan *machine learning*. Selain itu, *library* ini mudah untuk digunakan dan memiliki dokumentasi yang lengkap [20].

Kernel yang dipakai pada SVM adalah *linear kernel*, *polynomial kernel*, dan RBF *kernel*. Performa dari *kernel* beserta dengan *hyperparameters* yang dipakai akan dibandingkan menggunakan GridSearchCV. *Kernel* dan *hyperparameters* seperti nilai C dan nilai gamma dengan performa terbaik akan dipakai untuk melakukan *testing*.

3.1.6 Training, Testing, dan Evaluasi Model

Dataset akan dibagi menjadi 70% *training data* dan 30% *testing data* yang lalu akan digunakan untuk *training* dan *testing*. Lalu, model prediksi berita palsu akan dilatih menggunakan parameter terbaik dari hasil tahap sebelumnya. Dari hasil prediksi data *testing* akan dievaluasi keberhasilannya. Evaluasi dilakukan dengan bantuan *confusion matrix*. *Confusion matrix* dapat dilakukan perhitungan nilai akurasi, presisi, *recall*, dan F1-score dari model tersebut.

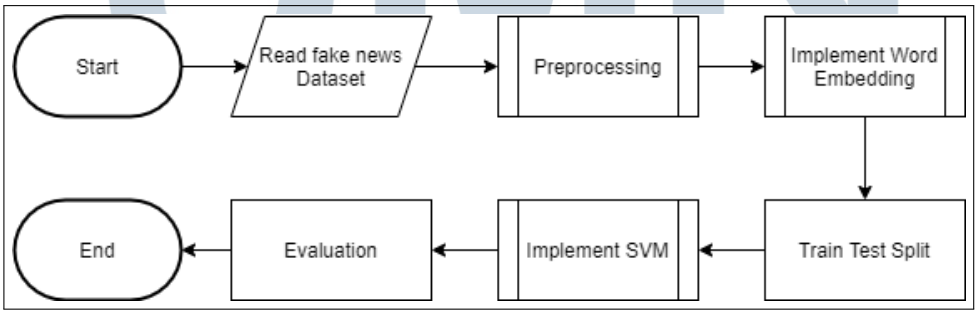
3.1.7 Penulisan Laporan

Hasil penelitian yang telah dilakukan lalu didokumentasikan dalam bentuk laporan. Laporan mencakup proses penelitian, perancangan, pembangunan sistem, hingga evaluasi model. Penulisan laporan akan dilakukan bertahap dan mengikuti sistematika penulisan yang sudah ditetapkan universitas.

3.2 Perancangan Sistem

3.2.1 Gambaran Umum

Dalam menjalankan penelitian ini diperlukan sebuah sistem untuk mengolah dataset untuk mencapai tujuan penelitian. Oleh karena itu, perancangan sistem dilakukan dalam bentuk menggambarkan langkah-langkah dalam melakukan penelitian ini. Langkah-langkah tersebut dilakukan menggunakan *flowchart* dan penjabaran dari awal hingga akhir.



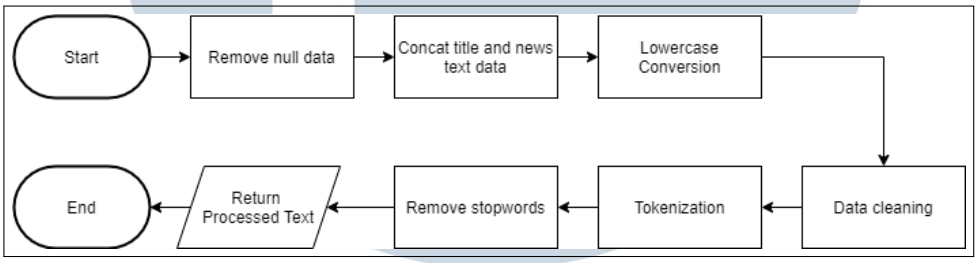
Gambar 3.2. Flowchart Gambaran Umum Perancangan Sistem

Pada gambar 3.2, perancangan sistem diawali dengan membaca dataset berita palsu. Dataset berita palsu dalam bentuk csv dan dapat dibaca menggunakan *function* read_csv dari *library* pandas. Selanjutnya, dilakukan *text preprocessing*, *train test*

split, implementasi *word embedding*, implementasi SVM, dan terakhir evaluasi sistem. Masing-masing sub-proses tersebut akan dijelaskan lebih detail pada sub-bab berikutnya.

3.2.2 Preprocessing

Preprocessing merupakan tahapan yang penting dilakukan untuk membersihkan dataset dari data-data yang tidak penting. Hal ini dilakukan sebelum menerapkan algoritma untuk memproses dataset. Preprocessing pada umumnya dilakukan pada dua jenis data yaitu data angka dan data teks. *Text preprocessing* meliputi *remove null data*, *lowercase coversion*, *Data cleaning*, *Tokenization*, dan *Remove Stopwords*



Gambar 3.3. Flowchart Text Preprocessing

Tahapan-tahapan yang terdapat pada gambar 3.3 dapat dijabarkan sebagai berikut:

1. *Remove null data*
   Tahapan yang umum dilakukan saat *preprocessing* baik data teks maupun data angka, yaitu menghilangkan entri data yang bernilai null.

2. *Concate title and news text data*
   Tahapan ini khusus dilakukan untuk dataset yang telah diambil. Pada dataset, judul dan isi dari teks berita dipisah sehingga perlu digabungkan terlebih dahulu sebelum diproses lebih lanjut.

3. *Lowercase conversion*
   Mengubah seluruh isi data teks menjadi lowercase. Hal ini dapat mengurangi jumlah kosakata dan kompleksitas data teks. Selain itu, *lowercase conversion* bermaksud untuk mengurangi ketersebaran data, karena sebuah kata memiliki arti yang sama walaupun berbeda *case*.

4. *Data cleaning*

   Tahapan ini menghapus unsur non-alphabet pada entri data menggunakan *regular expression* (regex). Unsur non-alphabet yang dihapus meliputi tanda baca, angka, karakter diakritik, dan karakter diluat kategori *latin-script alphabet*.

5. *Tokenization*

   Tahapan ini berfungsi untuk mengubah entri data teks menjadi daftar kata atau disebut dengan token.

6. *Remove stopwords*

   Menghapus kata yang termasuk kategori *stopwords* bahasa Inggris dari daftar kata. Kata yang termasuk kategori *stopwords* yaitu kata ganti, konjungsi, preposisi, dan kata-kata yang tidak dapat berdiri sendiri.

7. *Return processed text*

   Setelah semua proses *text preprocessing* sudah dilakukan maka data teks yang sudah bersih akan dikembalikan untuk diolah lebih lanjut.

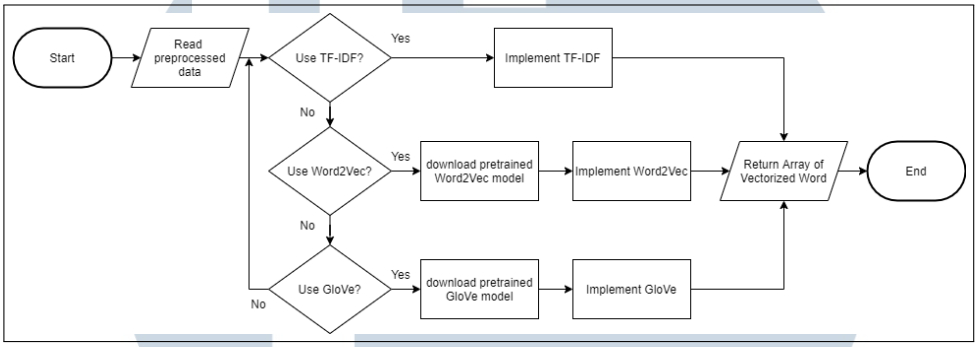
3.2.3 Implement Word Embedding

Sebelum memulai *training* algoritma SVM, data dalam bentuk teks perlu direpresentasikan kedalam bentuk angka atau vektor. Vektorisasi teks dapat dilakukan secara konvensional berbasis kemunculan kata dan algoritma *word embedding*. Pada penelitian kali ini menggunakan satu algoritma vektorisasi konvensional yaitu TF-IDF dan dua algoritma *word embedding* yaitu *GloVe* dan *Word2Vec*. Penggunaan tiga algoritma ini bertujuan untuk membandingkan metode vektorisasi kata yang digunakan lalu memilih metode terbaik untuk *classifier* SVM.

Pemilihan Word2Vec sebagai pembanding dengan GloVe bertujuan untuk menguji performa GloVe sebagai sesama *Word embedding* yang mampu menangkap arti kata. Sedangkan TF-IDF dipilih untuk menguji performa GloVe sebagai *word vectorizer* berbasis statistik. TF-IDF merupakan *word vectorizer* konvensional yang paling baik digunakan untuk diterapkan pada dataset besar dibandingkan *Count Vectorizer* (CV) [21]. Maka dari itu, perbandingan algoritma *feature extraction* dilakukan dengan TF-IDF walaupun TF-IDF bukan *word embedding*.

Pada gambar 3.4 untuk TF-IDF dapat langsung diterapkan pada data training dan testing, sedangkan untuk GloVe dan Word2Vec perlu memperoleh *pretrained model* terlebih dahulu sebelum dapat digunakan. Cara kerja TF-IDF adalah dengan

menghitung *term frequency* dan *inverse document frequency* sebuah kata, lalu kedua vektor tersebut dikalikan untuk mendapatkan vektor kata. Sedangkan *pretrained word embedding* bekerja dengan cara mengambil nilai vektor untuk kata tertentu dari kamus *pretrained model*. Setelah itu, vektor tersebut diratakan dimensinya sehingga mendapatkan hasil vektor kata tersebut.



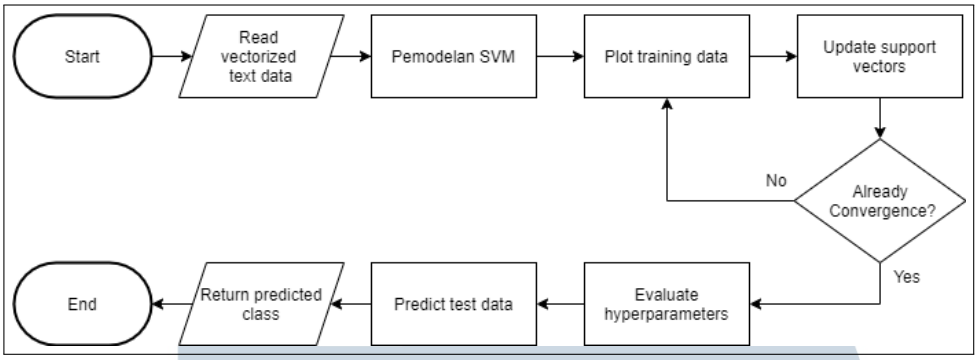
Gambar 3.4. Flowchart Implementasi Word Embedding

3.2.4 Train Test Split

Train test split termasuk dalam tahapan dalam membangun *supervised machine learning*. Pada tahapan ini, dataset yang sudah bersih akan dipisahkan menjadi data *training* dan data *testing*. Proporsi pembagian data *training* dan data *testing* pada penelitian ini adalah 70% dan 30%. Rasio yang dipilih merupakan rasio *train test split* paling optimal berdasarkan simulasi Monte Carlo [22]. Simulasi Monte Carlo adalah metode evaluasi performa model *machine learning* dengan iterasi proses *random sampling*. Data yang sudah dibagi kemudian akan digunakan untuk melatih SVM, lalu testing SVM untuk melihat keberhasilan pemodelan.

3.2.5 Implement SVM

Data teks yang sudah diubah menjadi data vektor akan diklasifikasikan menggunakan SVM. Pada gambar 3.5 setelah membaca data, SVM akan diberi *hyperparameter* untuk proses klasifikasi. Lalu SVM akan dilatih menggunakan data *training*. Proses *training* SVM berawal dari plot data *training*, lalu akan memulai iterasi untuk mencari *support vector* yang menghasilkan *hyperplane* pemisah antar kelas. Jika kondisi sudah konvergen maka iterasi akan berhenti. Setelah proses *training* selesai dilakukan maka model tersebut akan dipakai untuk melekaukan prediksi data *testing* dan mengembalikan kelas hasil prediksi.



Gambar 3.5. Flowchart Implementasi SVM

Keadaan konvergen adalah keadaan dimana SVM sudah menemukan *hyperplane* pemisah optimal yang menghasilkan solusi terbaik. *Hyperplane* yang optimal berbarti jarak *hyperplane* dengan titik-titik data terdekat masing-masing kelas sudah maksimal. Solusi terbaik dari SVM dapat diukur dengan *objective fucntion*. Fungsi ini dapat digunakan untuk mengukur taraf kesalahan SVM dalam melakukan klasifikasi. Sehingga SVM dapat menentukan kesalahan klasifikasi yang minimum. Selain kedua hal diatas SVM mencapai keadaan konvergen dengan kriteria tertentu, contohnya ketika sudah mencapai nilai maksimum iterasi [15].

Proses evaluasi *hyperparameter* dapat dilakukan dengan metode valdiasi silang (*cross validation*). Validasi silang dapat dilakukan dengan metode grid search. Metode ini bekerja dengan cara menerima beberapa input *hyperparameter* yang lalu akan dikombinasikan satu per satu. Setelah itu setiap kombinasi *hyperparameter* akan menghasilkan skor, lalu dari hasil skor tersebut akan dievaluasi dan diambil satu kombinasi *hyperparameter* dengan skor terbaik.

3.2.6 Evaluation

Tahap evaluasi adalah tahap terakhir dalam melakukan pemodelan *machine learning*. Evaluasi dilakukan untuk memastikan skor yang dihasilkan model atas hasil prediksi data *testing*. Evaluasi dapat dilakukan dengan bantuan *confusion matrix*. *confusion matrix* merupakan matriks yang berisikan jumlah kelas hasil prediksi dan kelas sebenarnya. *confusion matrix* dapat membantu untuk mengukur skor akurasi, presisi, *recall*, dan F1-score.