BAB III
METODOLOGI PENELITIAN

3.1. Subyek dan Obyek Penelitian

Subyek penelitian dari topik ini adalah metode *BLSTM-HMM*. Sedangkan untuk obyek penelitian dari topik ini adalah model *Named Entity Recognition* pada dokumen Twitter berbahasa Indonesia.

3.2. Alat dan Bahan Penelitian

Dalam kegiatan penelitian tentunya diperlukan sebuah alat dan juga bahan apa saja yang dibutuhkan dalam penelitian tersebut, tentunya bisa berupa perangkat keras, perangkat lunak, dan data penelitian.

3.2.1. Perangkat Keras

1. Komputer yang dibekali dengan prosesor AMD A8 7600 3.1 GHz, 8GB RAM, GPU GT 1030.
2. Koneksi internet.

3.2.2. Perangkat Lunak

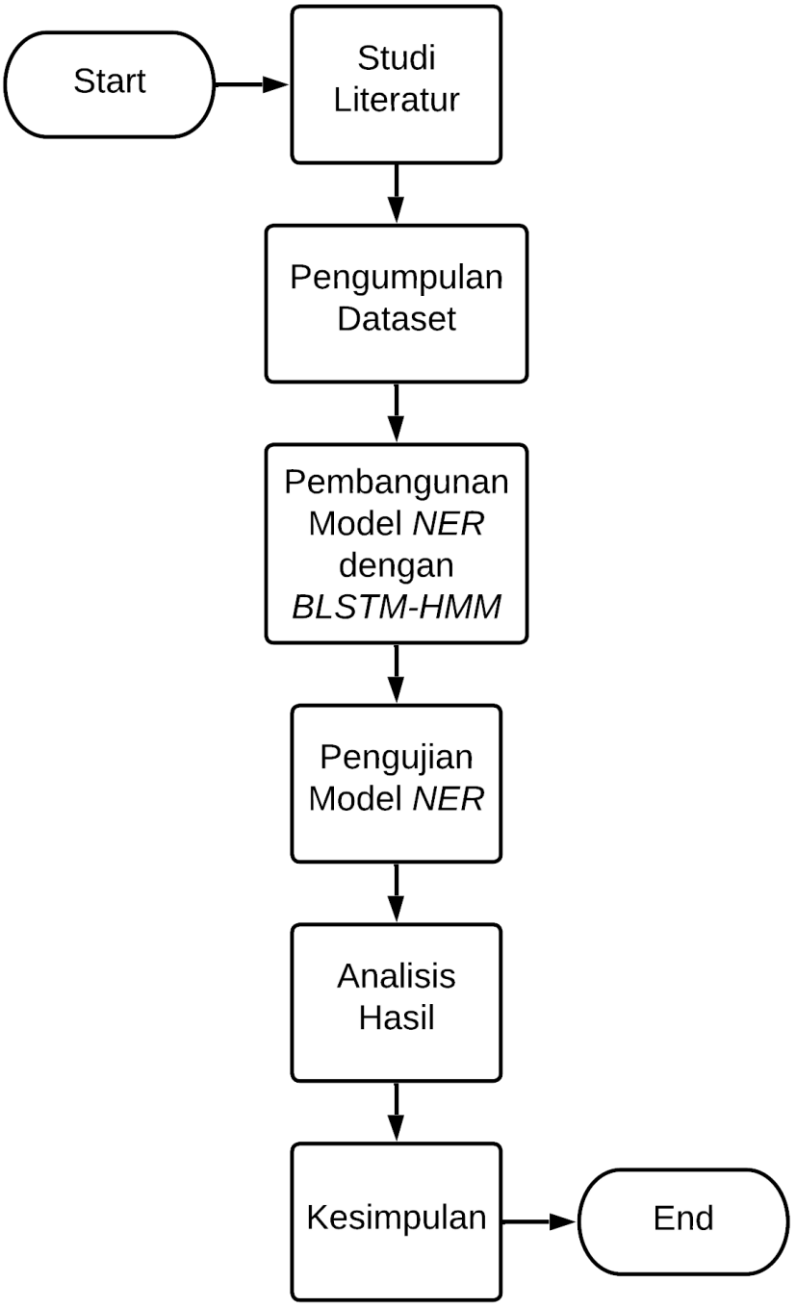
1. OS Windows 10 Pro 64-bit.
2. Browser.
3. Google Colaboratory.
4. Microsoft Office.

3.2.3. Data Penelitian

Data yang digunakan dalam penelitian ini mengambil data dari penelitian [6] yang berjumlah 500 tweets yang sudah dilakukan pelabelan dan *POS tag*. Data tersebut juga sudah terbagi menjadi data train dan test.

3.3. Diagram Alir Penelitian

Pada bagian ini, peneliti akan menjelaskan tahapan-tahapan penelitian yang akan dilakukan dalam membangun sebuah model *Named Entity Recognition* menggunakan metode *BLSTM-HMM* dengan fitur *FastText* dan *POS tag*. Tahapan-tahapan yang dilakukan di sajikan dalam Gambar 3.1.



Gambar 3.1 Diagram alir penelitian

3.3.1. Studi Literatur

Peneliti megeksplorasi dan mempelajari karya-karya literatur, seperti buku, jurnal, paper, artikel, atau dokumen ilmiah lainnya, dengan tujuan untuk mengumpulkan dan menganalisis sebanyak mungkin informasi yang terkait dengan topik yand diteliti, serta untuk mengetahui apa yang telah diketahui oleh para ahli tentang topik tersebut. Hasil dari studi literatur bisa dijadikan sebagai bahan rujukan untuk penelitian yang akan dilakukan oleh peneliti.

3.3.2. Pengumpulan Dataset

Data yang digunakan pada penelitian ini diambil dari penelitian [6] yang berjumlah 500 tweets yang sudah dilakukan pelabelan dan *POS tag*. Data tersebut juga sudah terbagi menjadi data train dan test. Data terdiri dari 7 label entity (*O, B-PER, I-PER, B-LOC, I-LOC, B-ORG, I-ORG*), dan terdiri dari 11 *POS tag* ("*NOUN*", "*PROPN*", "*VERB*", "*ADV*", "*ADP*", "*NUM*", "*PUNCT*", "*CONJ*", "*DET*", "*PRON*", "*ADJ*"). Gambaran umum data train dan data test dapat dilihat pada Tabel 3.1, dan 3.2.

Tabel 3.1 Preview Data Train

No.	**Kata**	***POS tag***	**Entity**
1.	Pengamat	*NOUN*	*O*
2.	politik	*NOUN*	*O*
3.	dari	*ADP*	*O*
4.	Universitas	*PROPN*	*B-ORG*
5.	Gajah	*PROPN*	*I-ORG*
6.	Mada	*PROPN*	*I-ORG*
7.	Arie	*PROPN*	*B-PER*
8.	Sudjito	*PROPN*	*I-PER*
9.	menilai	*VERB*	*O*
10.	Pengamat	*NOUN*	*O*

Tabel 3.2 Preview Data Test

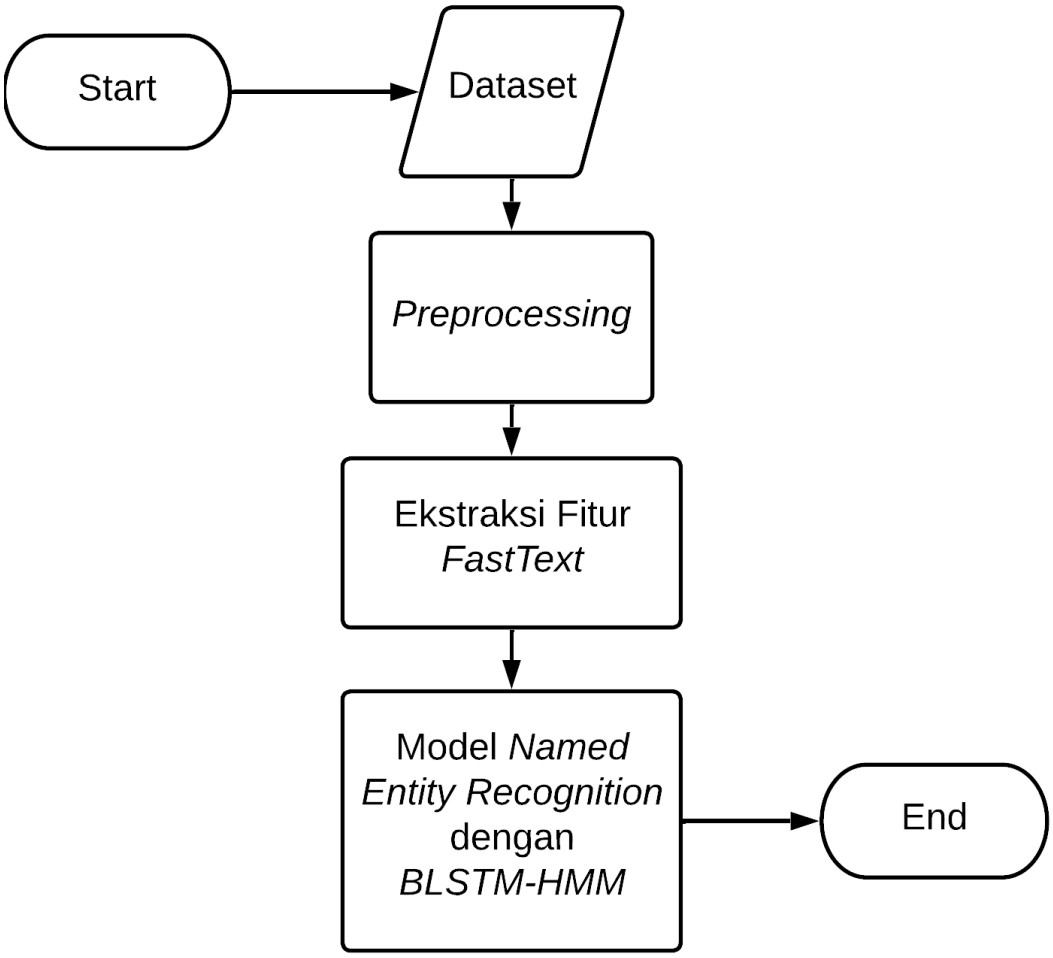
No.	Kata	*POS tag*	Entity
1.	Pengadilan	*PROPN*	*B-ORG*
2.	Negeri	*PROPN*	*I-ORG*
3.	Jakarta	*PROPN*	*I-ORG*
4.	Selatan	*PROPN*	*I-ORG*
5.	memutuskan	*VERB*	*O*
6.	penetapan	*VERB*	*O*
7.	tersangka	*NOUN*	*O*
8.	Budi	*PROPN*	*B-PER*
9.	tidak	*ADV*	*O*
10.	sah	*VERB*	*O*

Tabel 3.1 dan 3.2 preview data train dan data test di atas merupakan dataset train dan dataset test yang akan digunakan untuk melatih model *NER*. Data ini terdiri dari kata, *POS tag*, dan label entitas. Ada tiga jenis entitas yang dapat dikenali dan digunakan pada penelitian ini, yaitu Orang (*PER* - nama orang), Lokasi (*LOC* - lokasi atau nama area), dan Organisasi (*ORG* - nama organisasi). Data yang digunakan dilabeli dengan format *BIO* (*Begin, Inside, Outside*), dimana tag "B" mewakili awal dari suatu entitas, tag "I" digunakan untuk mewakili kata berikutnya dalam suatu entitas, dan tag "O" mewakili semua kata non-entitas lainnya [17]. Sedangkan *POS tagger* merupakan usaha untuk menetapkan label pada kata-kata dalam teks berdasarkan kelas kata dan mempertimbangkan potensi fitur morfologisnya [18].

3.3.3. Pembangunan Model *Named Entity Recognition*

Dalam pembangunan model *Named Entity Recognition* (*NER*) dapat dibagi menjadi beberapa tahapan. Dataset yang digunakan dalam penelitian ini mengambil

data dari penelitian [6] yang sudah terlabeli dan sudah di split antara data train dan test. Meskipun data yang digunakan sudah terlabeli, namun peneliti disini tetap menjelaskan tahap-tahap yang akan dilalui. Gambar 3.2 adalah gambaran tahapan dari sistem model *NER* yang akan dibangun pada penelitian ini.



Gambar 3.2 Desain sistem model *NER*

3.3.3.1. Data Preprocessing

Misi dari tahapan data preprocessing adalah mengubah data masukan awal ke dalam format yang sesuai untuk analisis berikutnya [19]. Preprocessing adalah langkah pertama dalam mengolah dataset untuk selanjutnya digunakan pada model. Tujuan dari preprocessing data adalah untuk memastikan bahwa data dalam bentuk yang baik dan dapat diolah oleh model dengan baik.

3.3.3.2. Ekstraksi Fitur

Penelitian ini menggunakan data dari penelitian [6] yang telah diberi *POS tag*. Pentingnya ekstraksi fitur termanifestasikan dalam klasifikasi teks, di mana tujuannya adalah mengubah format teks yang bersifat tidak terstruktur menjadi struktur, memungkinkan pengolahan oleh algoritma *Machine Learning* guna melakukan klasifikasi ke dalam kelas yang telah ditetapkan sebelumnya [18].Fitur yang digunakan dalam penelitian ini adalah *Part of Speech (POS) tag* yang didapat dari data penelitian [6], dan penambahan *Word embedding FastText*.

Word embedding adalah cara untuk merepresentasikan kata-kata sebagai vector numerik, yang kemudian dapat digunakan sebagai masukan untuk model *Machine Learning*. *Word embedding* merupakan konsep yang merujuk pada teknik mengonversi sebuah kata menjadi suatu vektor atau array yang terdiri dari sejumlah angka [20]. *Word embedding* menangkap konteks di mana sebuah kata muncul dalam kumpulan teks, dan hubungan antar kata. Ini berguna karena memungkinkan model *Machine Learning* memperhitungkan arti kata, bukan hanya karakter atau kata itu sendiri.

FastText adalah toolkit yang dikembangkan oleh tim riset Facebook untuk pembelajaran yang efektif dari representasi kata dan klasifikasi teks [21]. *FastText* mengasumsikan sebuah kata yang akan disusun oleh *n-gram* karakter di mana panjang dari *n* dapat berubah dari satu sampai ke panjang kata tersebut [21]. Pada penelitian kali ini menggunakan subword trigram. Keunggulan dari penerapan *FastText* terletak pada kemampuannya untuk menyimpan vektor kata sebagai n-gram karakter, memungkinkan identifikasi representasi vektor untuk kata-kata yang tidak langsung terdapat dalam kamus [21]. Maka cara kerja *FastText* dapat dilihat yaitu:

1. Input layer, input pada *FastText* adalah sekumpulan kata yang berasal dari suatu kalimat yang sudah dipecah menjadi kata atau token, proses ini sama seperti *tokenization*.
2. *Embedding* layer, pada tahap ini setiap kata akan dipecah menjadi karakter subword (*n-gram*). Kemudian setiap karakter subword akan diubah menjadi vektor menggunakan *one-hot encoding*. *One-hot encoding* adalah teknik

representasi data yang umum digunakan dalam bidang pemrosesan bahasa alami dan pembelajaran mesin [1]. Dalam penelitian ilmiah, *one-hot encoding* digunakan untuk mengkonversi variabel kategori menjadi representasi vektor biner yang disebut vektor "*one-hot*" [1]. Setiap kategori dalam variabel kategori diwakili oleh satu vektor, di mana panjang vektor sama dengan jumlah kategori yang ada dalam data. Vektor tersebut terdiri dari nilai nol kecuali pada posisi yang sesuai dengan kategori yang diwakili, di mana nilainya adalah satu. Misalnya, jika terdapat tiga kategori ("A", "B", dan "C"), representasi one-hot encoding untuk kategori "A" akan menjadi [1, 0, 0], untuk kategori "B" akan menjadi [0, 1, 0], dan untuk kategori "C" akan menjadi [0, 0, 1]. Pada tabel 3.6 dijelaskan contoh sederhana pada *embedding* layer.

Tabel 3.3 Contoh Embedding Layer Tabel

Kata / Token	**Subword (*n-gram*)**	**Vektor per-Subword**
Joko	"Jo", "Jok", "oko"	[1,0,0],[0,1,0],[0,0,1]
Widodo	"Wi", "Wid", "ido", "dod", "odo"	[1,0,0,0,0],[0,1,0,0,0],[0,0,1,0,0],[0,0,0,1,0],[0,0,0,0,1]
tidak	"ti", "tak", "tid", "ida", "dak"	[1,0,0,0,0],[0,1,0,0,0],[0,0,1,0,0],[0,0,0,1,0],[0,0,0,0,1]
menghadiri	"me", "men", "eng", "nga", "gah", "aha", "had", "adi", "dir", "iri"	[1,0,0,…,0],[0,1,…,0,0],[0,0,1,…,0],[0,0,0,1,…,0],[0,0,0,0,1,…,0] dst.
secara	"se", "sec", "eca", "car", "ara"	[1,0,0,0,0],[0,1,0,0,0],[0,0,1,0,0],[0,0,0,1,0],[0,0,0,0,1]
langsung	"la", "lan", "ang", "nga", "gsu", "sun", "ung"	[1,0,0,…,0],[0,1,…,0,0],[0,0,1,…,0],[0,0,0,1,…,0],[0,0,0,0,1,…,0] dst.
sidang	"si", "sid", "ida", "dan", "ang"	[1,0,0,0,0],[0,1,0,0,0],[0,0,1,0,0],[0,0,0,1,0],[0,0,0,0,1]

Kata / Token	Subword (*n-gram*)	Vektor per-Subword
PBB	"PB", "PBB"	[1,0],[0,1]
yang	"ya", "yan", "ang"	[1,0,0],[0,1,0],[0,0,1]
digelar	"di", "dig", "ige", "gel", "ela", "lar"	[1,0,0,…,0],[0,1,…,0,0],[0,0,1,…,0],[0,0,0,1,…,0],[0,0,0,0,1,…,0] dst.
di	"di"	[1]
New	"Ne", "New"	[1,0],[0,1]
York	"Yo", "Yor", "ork"	[1,0,0],[0,1,0],[0,0,1]

Vektor *one-hot encoding* memiliki panjang yang sama dengan ukuran kosakata dan memiliki nilai 1 pada posisi indeks yang sesuai dengan karakter atau subword yang diwakili, sementara nilai-nol pada posisi lainnya. Vektor ini menciptakan representasi biner dari setiap subword yang dapat digunakan sebagai input untuk model *word embedding*, termasuk model *FastText*.

3. *Average pooling*, yaitu vektor karakter subword dijumlahkan dan diambil rata-ratanya untuk menghasilkan vektor representasi kata. Proses ini membantu memperhitungkan informasi dari semua karakter subword dan memberikan representasi yang lebih kaya. Bisa dilihat pada Tabel 3.7.

Tabel 3.4 Contoh *Average Pooling*

Subword (*n-gram*)	Vektor per-Subword	*Average Pooling*
"Wi", "Wid", "ido", "dod", "odo"	[1,0,0,0,0],[0,1,0,0,0],[0,0,1,0,0],[0,0,0,1,0],[0,0,0,0,1]	[0.2, 0.2, 0.2, 0.2, 0.2]

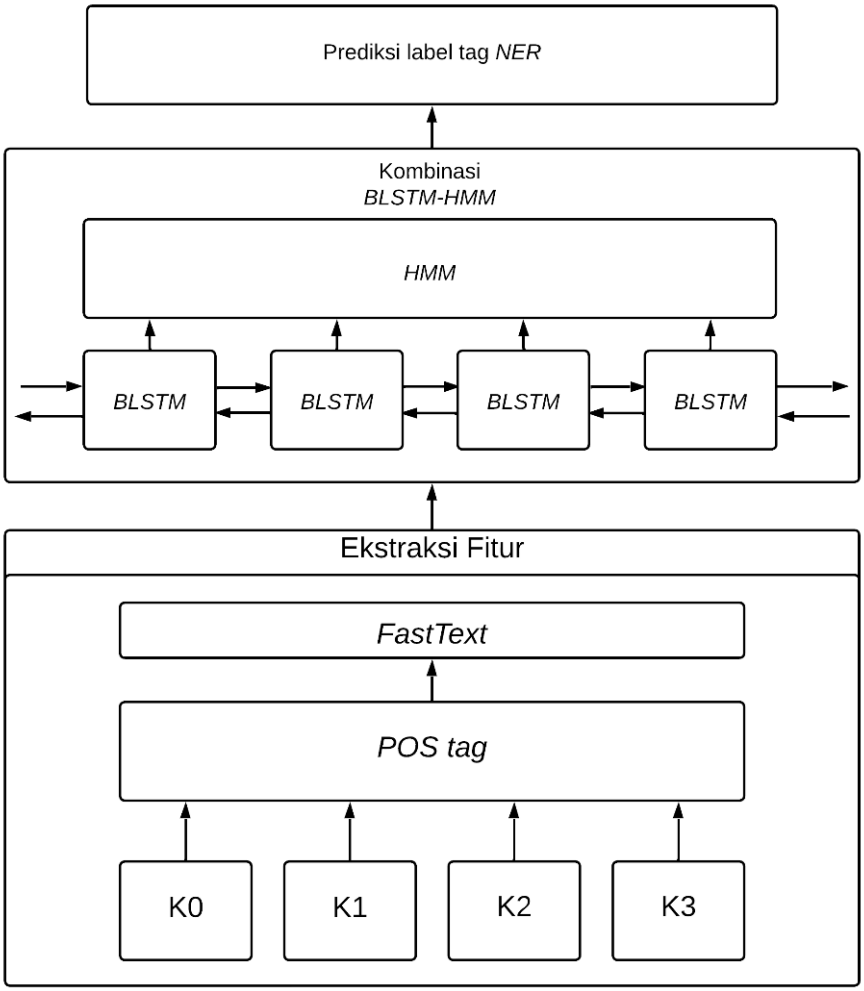
Jumlahkan setiap vektor per-subword, setelah itu bagi hasil penjumlahan vektor per-subword dengan jumlah subword yang ada.

4. Output layer, vektor representasi kata yang diperoleh diumpankan ke dalam model pembelajaran mesin untuk dilatih. Parameter model, termasuk vektor

representasi kata, akan diperbarui selama pelatihan berlangsung. Setelah pelatihan model dapat digunakan untuk berbagai tugas *NLP* seperti klasifikasi teks, pemodelan bahasa, dan pemrosesan teks terkait lainnya.

3.3.3.3. Model NER dengan BLSTM-HMM

Proses membangun model *NER* dengan metode *BLSTM-HMM* ini dilakukan setelah proses preprocessing, data anotasi, ekstraksi fitur. Pada tahapan ini dilakukan proses model training menggunakan metode *BLSTM-HMM*. Arsitektur *BLSTM-HMM* dapat dilihat pada Gambar 3.3.



Gambar 3.3 Arsitektur *BLSTM-HMM*

Dataset teks yang telah di *tokenization* dan sudah dilabeli akan dilakukan ekstraksi fitur yaitu, pemberian *POS tag* dan selanjutnya masing-masing kata tersebut direpresentasikan menggunakan vektor hasil dari *FastText*. Vektor representasi kata hasil *FastText* dari setiap kata sebelumnya dimasukkan sebagai input ke lapisan *BLSTM*. Lapisan *BLSTM* akan memproses urutan ini secara sekuensial. *BLSTM* melakukan proses secara simultan ke depan (*forward*) dan ke belakang (*backward*) melalui urutan kata yang memungkinkan model untuk memahami konteks sepanjang urutan. Output dari lapisan *BLSTM*, baik dari arah depan maupun belakang, dihasilkan sebagai fitur ekstraktif dari urutan kata. Output dari kedua arah (*forward* dan *backward*) akan digabungkan, misalnya dengan menggabungkan nilai dari setiap langkah waktu. Output dari *BLSTM* dapat diproses melalui lapisan *dense* layer dengan aktivasi *softmax* untuk menghasilkan label entitas. Selanjutnya, inisialisasi model *HMM* dengan memperhitungkan jumlah state dimana state mewakili label entitas. Setelah itu menghitung probabilitas emisi untuk setiap state berdasarkan output *BLSTM*, dimana probabilitas emisi dapat diasosiasikan dengan seberapa cocok output *BLSTM* dengan state tertentu. Kemudian, menghitung probabilitas transisi antar state berdasarkan frekuensi transisi dari output *BLSTM*, ini memodelkan seberapa sering model melihat transisi antara entitas tertentu. Gunakan output *BLSTM* sebagai observasi atau input ke model *HMM* dimana setiap observasi sesuai dengan label entitas yang ada. Terakhir, mendekode urutan state yang paling mungkin berdasarkan observasi dari output *BLSTM*. Berdasarkan langkah tersebut, mengintegrasikan kemampuan model *BLSTM* dalam memahami konteks sekuensial, dengan kemampuan model *HMM* dalam memodelkan transisi antar label entitas sehingga menghasilkan output label *NER*.

3.3.4. Pengujian Model NER

Pengujian model *NER* menggunakan *BLSTM-HMM* dengan fitur *POS tag* dan *FastText* melibatkan penerapan model yang telah dihasilkan selama tahap pembangunan pada kumpulan data test yang telah disiapkan. Di dalam proses ini, model akan diterapkan untuk mengidentifikasi entitas bernama, seperti orang,

lokasi, dan organisasi, yang terdapat dalam teks yang ada dalam data test. Dengan demikian, pengujian bertujuan untuk mengevaluasi sejauh mana model mampu mengenali dan membedakan entitas bernama dari data test yang diberikan.

3.3.5. Analisis Hasil

Setelah melakukan proses pembangunan model *NER*, dalam analisis hasil akan membahas evaluasi kinerja dari model *NER* dengan menggunakan metrik F1 score, yang merupakan pengukuran yang menggabungkan *presisi* dan *recall* untuk memberikan pemahaman yang lebih menyeluruh tentang keefektifan model dalam mengidentifikasi entitas bernama. Selain itu, akan dilakukan perbandingan kinerja model NER yang dikembangkan dengan penelitian [6] sebelumnya untuk mengevaluasi apakah peningkatan atau perubahan metode yang diusulkan memberikan hasil yang lebih baik atau tidak.

3.3.6. Kesimpulan

Setelah menyelesaikan semua tahapan diagram alir, Peneliti akan menyimpulkan hasil penelitian yang sudah dilakukan. Kesimpulan dari analisis hasil penelitian yang diperoleh dari model *NER* yang sudah dibuat, dan kesimpulan dari hasil karya tulis ilmiah. Dari hasil kesimpulan tersebut diharapkan dapat membantu penelitian-penelitian selanjutnya.